Arc Routing

Problems, Methods, and Applications

Edited by

Ángel Corberán
University of Valencia
Burjassot, Valencia
Spain

Gilbert Laporte
HEC Montréal
Montréal, Quebec
Canada

Society for Industrial and Applied Mathematics
Philadelphia

Mathematical Optimization Society
Philadelphia

10 9 8 7 6 5 4 3 2 1

(continued)

Chapter 1

A Historical Perspective on Arc Routing

Ángel Corberán
Gilbert Laporte

1.1 ▪ Introduction

The study of arc routing is rooted in the seminal work of the Swiss mathematician Leonhard Euler who was called to study the famous Königsberg bridges problem in the 18th century while he was chair of mathematics at the St. Petersburg Academy of Sciences. The study of this problem led Euler to lay the foundation for modern graph theory. Recent accounts of the scientific career of Euler and of the history of the bridges of Königsberg are provided in Assad [2] and in Gribkovskaia, Halskau, and Laporte [20], respectively. A number of 19th century mathematicians also showed some interest in arc routing problems, namely, in relation to feasibility problems arising in graph traversals. However, the study of modern arc routing truly started in 1960 with the first publication on the Chinese postman problem. Over the following years and to this day, arc routing has evolved into a rich research area, firmly embedded within the broader domain of combinatorial optimization. This introductory chapter identifies the pioneer researchers in the field of arc routing and some of its main modern contributors.

1.2 ▪ Origins

We will first summarize some of the early works related to graph traversals. In particular, the Königsberg bridges problem is revisited and some comments on figure drawing and maze traversal are also included. The reader is referred to the excellent book by Biggs, Lloyd, and Wilson [6] for more details on these fascinating topics.

1.2.1 ▪ The Königsberg bridges problem

"In Königsberg in Prussia, there is an island A, called *the Kneiphof*; the river which surrounds it is divided into two branches, as can be seen in the figure (see Figure 1.2), and these branches are crossed by seven bridges a, b, c, d, e, f, and g. Concerning these bridges, it was asked whether anyone could arrange a route in such a way that he would

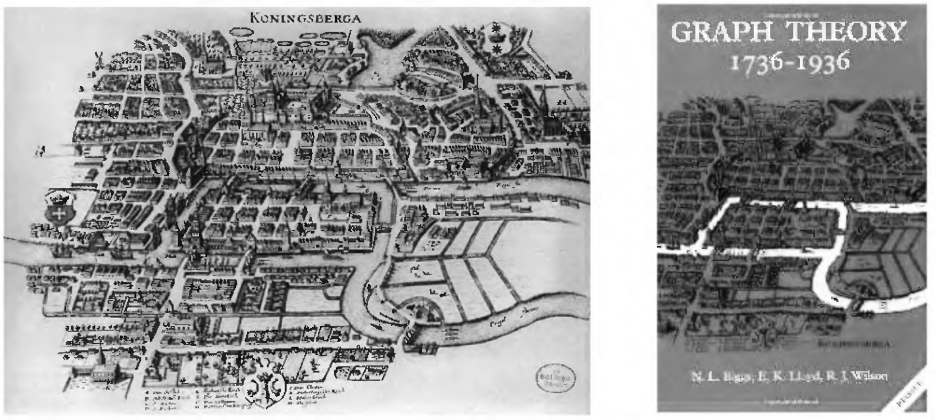


Figure 1.1. *Königsberg in the 18th Century. Front cover of the Biggs, Lloyd, and Wilson book (Oxford University Press).*

cross each bridge once and only once. I was told that some people asserted that this was impossible, while others were in doubt; but nobody would assert that it could be done."

This is how Euler [16] described the well-known Königsberg bridges problem (see Biggs, Lloyd, and Wilson [6] for a full English translation of the original paper).

And Euler continued: "From this, I have formulated the general problem: whatever be the arrangement and division of the river into branches, and however many bridges there be, can one find out whether or not it is possible to cross each bridge exactly once?"

Note that Euler wrote about finding a route traversing all the bridges exactly once, but it is not yet clear whether he was asking about the existence of a "closed" or of an "open" route. While some authors formulate the Königsberg bridges problem as that of finding an open route crossing each bridge exactly once (see, for instance, Assad and Golden [3], Biggs, Lloyd, and Wilson [6], Hopkins and Wilson [25], and Grötschel and Yuan [21]), others also require that the route should return to the starting point (see Wilson and Watkins [36], Eiselt, Gendreau, and Laporte [14], Zaragoza [38], Ahr [1], Gribkovskaia, Halskau, and Laporte [20], and Wøhlk [37]).

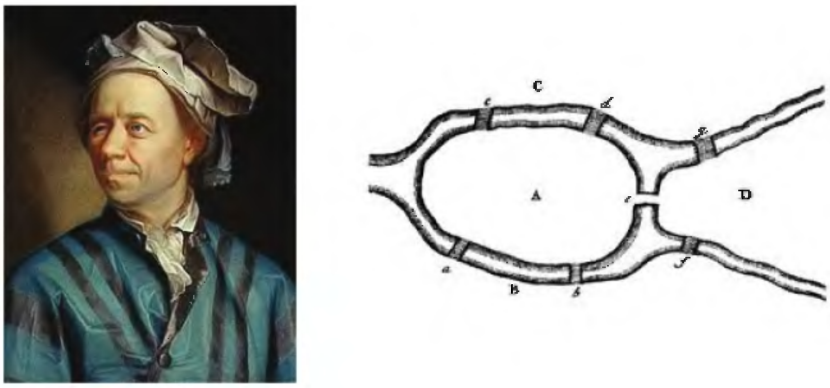


Figure 1.2. *Leonard Euler (1707–1783) and his drawing of the river Pregel and the seven bridges.*

At the end of [16], Euler pointed out: "So whatever arrangement may be proposed, one can easily determine whether or not a journey can be made, crossing each bridge once, by the following rules. If there are more than two areas to which an odd number of bridges lead, then such journey is impossible. If, however, the number of bridges is odd for exactly two areas, then the journey is possible if it starts in either of these areas. If, finally, there are no areas to which an odd number of bridges leads, then the required journey can be accomplished starting from any area."

It is now clear that Euler used the term "route" in a broad sense and interpreted the problem as that of finding a walk, closed or not, traversing each bridge exactly once. However, the "closed" version of the problem has an important advantage because of its direct relationship with Eulerian graphs. Recall that a connected graph G is said to be Eulerian (or unicursal) if there exists a closed walk including all the edges of G exactly once. Therefore, the closed version of the problem can be stated as follows: Is the graph representing the bridges of Königsberg Eulerian? Since the four vertices of the graph representing the Königsberg areas have odd degrees, the answer is no, and there is no open or closed route crossing exactly once each of the seven bridges of Königsberg.

1.2.2 ▪ Drawing figures

A closely related problem is that of drawing a given figure with the minimum number of strokes. In 1809, Poinsot [33] used arguments similar to those of Euler to prove that figures consisting of n points, where each point is joined to every other point, can be drawn in only one stroke if n is odd, but not if n is even. Note that these figures (see Figure 1.4) correspond to what we call complete graphs, and complete graphs are Eulerian for odd values of n.

Figure 1.3. *Louis Poinsot (1777–1859) and Johann Benedict Listing (1808–1882).*

In 1847, Listing, first a student and later a collaborator of Gauss, wrote a book on topology [28] which included some parts about figure drawing. Among other interesting comments, he pointed out that Figure 1.5a has eight odd-degree vertices and cannot therefore be drawn with less than four lines. In a sense, this sentence contains the key idea for solving what will later become known as the Chinese postman problem: to link the odd-degree vertices of the graph.

Finally, let us mention that not so many years ago, schoolchildren from many countries used to play a game consisting of drawing some figures with just one stroke. For instance, it is mentioned at http://www.mathematische-basteleien.de/house.html that the

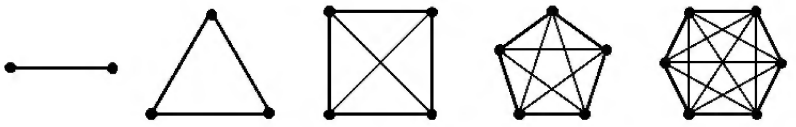
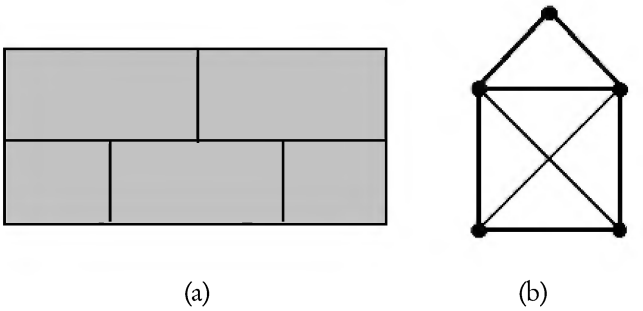


Figure 1.4. *Complete graphs with two to six vertices.*

House of Santa Claus is a very old German drawing game for small children: "You have to draw a house (see Figure 1.5b) in one line. You must not lift your pencil while drawing. You must not repeat a line. During drawing you have to say "Das ist das Haus des Nikolaus (This is Santa Claus' house)." You must speak a syllable of this sentence to every straight line. Girls must know "Wer dies nicht kann, kriegt keinen Mann (Who can't do this drawing, will get no man)."



(a) (b)

Figure 1.5. *Listing's drawing and the house of Santa Claus.*

1.2.3 ▪ Mazes and labyrinths

At the end of 19th century, several researchers became interested in the problem of escaping from a maze. In his book *Theorie der endlichen und unendlichen Graphen*, König [26] devotes a chapter to this problem (*Das Labyrinthenproblem*), in which he discusses in detail the algorithms of Wiener, Trémaux, and Tarry. König states the problem as follows: "A labyrinth can be identified with a graph, more precisely a finite and connected graph; the vertices of this graph correspond in the labyrinth to the branch points and the endpoints of the blind alleys."

"It is now required to give a method for reaching a specified point, which is usually treated as the center of the labyrinth. Since it is entirely arbitrary which point is treated as the center, the problem is only to be solved by specifying a way to reach every point of the labyrinth. If the plan of the whole labyrinth is known, there is no difficulty."

And König continues: "We will first assume that if during the journey one arrives at a vertex P, it is known for every edge ending at P whether it has already been traversed; besides this we assume that (like Theseus with the aid of Ariadne's thread) one can at any moment retrace in the opposite direction the whole of the path already covered (with the same repetitions). Under these assumptions the problem was solved by Wiener, who was the first ever to treat the age-old problem of the labyrinth as a mathematical one."

Figure 1.6. *Left: Hever castle. Right: Hampton Court maze.*

Figure 1.7. *Gaston Tarry (1843–1913) and Dénes König (1884– 1944).*

A description of the method proposed by Wiener follows, as well as some comments on it: "The method of Wiener cannot be called economical, since in general the edges have to be walked through a great many times. This is the more noticeable in that there is a journey through all the edges of the graph in which no edge is traversed more than twice."

Note that if every edge in the graph representing the labyrinth is duplicated, one obtains another graph with even-degree vertices. Since this graph is Eulerian, there exists a closed walk traversing every edge of the original graph twice, although, as pointed out by Biggs, Lloyd, and Wilson [6], this is only an argument proving the existence of a solution, and not a practical algorithm.

As noted by König [26], Trémaux proposed an algorithm which assumed that "on arrival during the journey at any vertex P, it is known for every edge ending at P whether it has already been walked, and if so how many times (the directions in which the edges have been walked are not assumed to be known here)." However, it was Tarry [34] who proposed the simplest procedure to escape from a labyrinth, just assuming that "every time we arrive at any vertex P it is known for every edge PK ending at P whether it has been traversed in the direction of K; and we can always determine the edge by which P

was reached for the first time (the "entry edge" of P)." The method proposed by Tarry guarantees that every edge is traversed at most once in each direction. It works as follows: "If you come via an edge PQ to a vertex Q, continue the journey along any edge QR that has not yet been traversed in the direction of R; but do not choose QR to be the entry edge of Q unless all other edges QK ending at Q have already been traversed in the direction of K."

1.3 ▪ Characterizations of Eulerian graphs

We now describe the necessary and sufficient conditions an undirected, directed, or mixed graph G has to satisfy in order to be Eulerian. Obviously, undirected graphs need to be connected, whereas directed and mixed graphs need to be strongly connected. In what follows we assume that these conditions hold.

1.3.1 ▪ Undirected Eulerian graphs

Euler stated necessary conditions for an undirected graph G to be Eulerian. He also pointed out they are sufficient without actually providing a proof of it. Sufficiency was proved in 1873 in a posthumous paper by Hierholzer [24]. Hence,

An undirected connected graph G is Eulerian if and only if every vertex of G has even degree.

At the end of his celebrated paper, Euler mentions the problem of constructing a Eulerian tour if one exists: "When it has been determined that such a journey can be made, one still has to find how it should be arranged. ... I do not think it worthwhile to give any further details concerning the finding of the routes." One can therefore assume that Euler thought this was a trivial issue. It was also Hierholzer [24] who proposed a linear-time algorithm to find a Eulerian walk in an undirected Eulerian graph G. This algorithm is described in Chapter 2.

A second, and very interesting, characterization of undirected Eulerian graphs was given in 1912 by Veblen [35]:

An undirected connected graph G is Eulerian if and only if it is the disjoint union of some cycles.

1.3.2 ▪ Directed Eulerian graphs

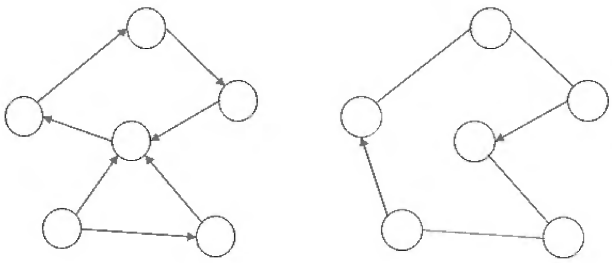
Clearly, the evenness of the vertices of a directed graph $G = (V,A)$ is again a necessary condition for the graph to be Eulerian. However, as can be seen in Figure 1.8a, which shows an even but non-Eulerian graph, it is not sufficient. A sufficient condition for a directed graph G to be Eulerian was given by König [26]: Every vertex of G must be symmetric; i.e., the number of arcs entering and leaving each vertex must be equal. Hence,

A directed and strongly connected graph G is Eulerian if and only if every vertex of G is symmetric.

Hierholzer's algorithm for undirected graphs can be easily adapted to determine a Eulerian walk of a directed Eulerian graph.

1.3.3 ▪ Mixed Eulerian graphs

A vertex of a mixed graph $G = (V, E, A)$ is said to have even degree if it is incident to an even number of edges and arcs. Again, the evenness of the vertices of G is a necessary but not a sufficient condition for a mixed graph to be Eulerian. However, it is easy to see that evenness together with symmetry are sufficient conditions for unicursality. But are evenness and symmetry also necessary conditions for a mixed graph to be Eulerian? The answer is obviously no. For instance, the mixed graph shown in Figure 1.8b is not symmetric although is clearly an Eulerian graph.



(a) (b)

Figure 1.8. *Non-Eulerian directed graph and Eulerian mixed graph.*

Figure 1.9. *Lester Randolph Ford Jr. and Delmer Ray Fulkerson (1924– 1976).*

The next question is then, "Are there necessary and sufficient conditions for a mixed graph G to be Eulerian?" Ford and Fulkerson answered this question in 1962 in their celebrated book *Flows in Networks* [17]: Every vertex of G must be even and graph G must be balanced, i.e., for every subset of vertices $S \subseteq V$, the difference between the number of arcs leaving S and the number of arcs entering S must be less than or equal to the number of edges incident with S. This last condition is generally known as the "balanced-set condition." Hence,

A mixed and strongly connected graph G is Eulerian if and only if G is even and balanced.

Figure 1.10 shows two even graphs. The first one does not satisfy the balanced-set condition and is not Eulerian, whereas the second one is a balanced and therefore Eulerian graph.

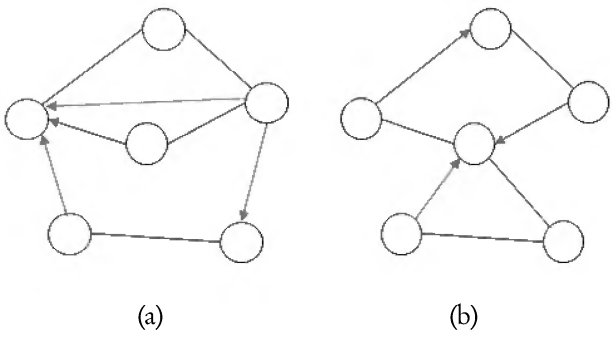


Figure 1.10. *Unbalanced and balanced mixed graphs.*

Note that checking whether a mixed graph G satisfies the balanced-set condition is a more complex task than checking whether it is even or symmetric since a condition must be verified for every subset of vertices. Fortunately, Ford and Fulkerson [17] also proposed an efficient algorithm which, applied to an even graph, either transforms the mixed graph into a completely directed symmetric (and therefore Eulerian) graph or fails, in which case the graph is not balanced (nor Eulerian). Let us also mention here that Nobert and Picard [30] presented a polynomial-time procedure that identifies a violated balanced-set inequality in a mixed graph if one exists (see also Benavent, Corberán, and Sanchis [5] for an explicit description and a proof of the procedure).

It is interesting to note that the balanced-set condition applied to a mixed graph without any edge (a directed graph) reduces to the symmetry condition of each subset of vertices, a condition that is satisfied if all the vertices are symmetric. Therefore, the necessary and sufficient conditions for a directed graph to be Eulerian can be obtained from the corresponding conditions for mixed graphs.

Finally, as was observed by Zaragoza [38], the characterization of Veblen for undirected Eulerian graphs in terms of cycle decompositions also applies to directed and mixed graphs.

1.4 ▪ The emergence of optimization

The Königsberg bridges problem posed the question of whether there exists a closed walk traversing each of the bridges exactly once. In general terms, the question is "Given an undirected and connected graph G, is there a closed walk traversing all its edges exactly once?" The answer to this decision problem is "Yes" or "No." There is no optimization involved in it.

It was Meigu Guan (Mei-Ko Kwan) who in 1960 introduced optimization to the problem. Guan is a Chinese mathematician who was working at that time at the Shandong Normal University. It is commonly assumed that Guan's work originated when he was working at a post office during the Chinese Cultural Revolution. However, according to Grötschel and Yuan [21], it seems that like many other scientists in China, Guan was encouraged to solve real-life problems during the Great Leap Forward movement (1958–1960), which attempted to transform the country from an agrarian to a modern economy.

The problem stated by Guan in a paper published in Chinese in 1960 and in English in 1962 [22] is as follows:

When the author was plotting a diagram for a mailman's route, he discovered the following problem: "A mailman has to cover his assigned segment before returning to the post office. The problem is to find the shortest walking distance for the mailman."

Figure 1.11. *Meigu Guan. Courtesy of Ya-Xiang Yuan [21].*

This problem is widely known as the Chinese postman problem (CPP). It seems that Alan Goldman mentioned it to Jack Edmonds when Edmonds was a member of Goldman's operations research group created in 1961 at the US National Bureau of Standards. It is not known whether Goldman suggested the name "Chinese postman problem" to Edmonds or whether it was Edmonds who coined the name. However, it seems to have first appeared in the title of an abstract by Edmonds [11] for the 27th National Meeting of the Operations Research Society of America (May 1965): "The Chinese Postman's Problem."

The CPP can also be interpreted as the problem of finding a subset of edges with minimum length which, added to the original graph, produces a Eulerian graph. Guan noted that a graph has an even number of odd-degree vertices and that a Eulerian graph can be obtained by replicating some edges in order to connect the odd-degree vertices. Based on this observation, Guan [22] proposed an algorithm for the CPP. Unfortunately, Guan's procedure is nonpolynomial.

A problem closely related to the CPP is the generalized Königsberg bridges problem, studied by Bellman and Cooke [4]: "In connection with a given graph, we ask for the paths which traverse every edge at least once and for which the number of repetitions of edges is a minimum." Also in this case, the procedure proposed by the authors was nonpolynomial: "The (dynamic programming) procedure outlined can require a large number of comparisons. The number is difficult to estimate."

In the above-mentioned abstract (see [11]), Edmonds writes "We present an algorithm ... (that) is *good* in the sense that the amount of work in applying it is at worst moderately algebraic, relative to the size of the graph, rather than exponential. It combines two earlier known algorithms: (1) the well-known *shortest path* algorithm, (2) a recent algorithm for *maximum matching*." Of course, Edmonds was referring to his celebrated polynomial algorithm for the minimum cost matching problem [12], and the procedure for the CPP

Figure 1.12. *Alan J. Goldman (1933–2010) and Jack Edmonds.*

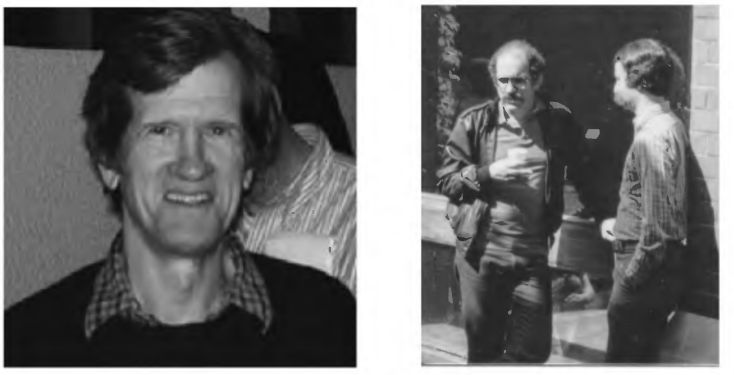


Figure 1.13. *Ellis L. Johnson. Nicos Christofides and Bruce L. Golden.*

mentioned consists of constructing first a complete graph G' whose vertices are the odd-degree vertices of the original graph G and whose edge costs are those of shortest paths linking them in G, and computing a perfect matching of minimum cost in G'.

The full details and many other outstanding results, including the complete description of the CPP polyhedron and several algorithms for the directed and mixed versions of the CPP, were published in 1973 in a seminal paper coauthored with Ellis Johnson [13]. Also in 1973, Christofides [8] proposed the same two-step procedure to solve the CPP.

While the directed version of the CPP can also be solvable in polynomial time (Edmonds and Johnson [13]), the CPP defined on a mixed graph (MCPP) is NP-hard. It was Papadimitriou [32] who proved this important result in 1976. However, Edmonds and Johnson proposed a polynomial-time algorithm for the MCPP defined on even graphs. Based on this algorithm, Frederickson [18] proposed two heuristics, each having a worst-case ratio of 2 for this problem, although applying both heuristics and selecting the best solution leads to a worst-case ratio of 5/3. Since the CPP defined on a windy graph (Minieka [29]) contains the MCPP as a special case, it is also an NP-hard problem (Brucker [7], Guan [23]).

In closing this section, we mention two important extensions of the CPP. In the first, called the Rural Postman Problem (RPP), some arcs or edges of the graph require service


Figure 1.14. *Christos H. Papadimitriou and Greg N. Frederickson.*


Figure 1.15. *Larry D. Bodin and Bruce L. Golden.*

while the others do not but may be traversed in the solution. The aim is to determine a minimum cost traversal of the required arcs and edges of the graph. This problem was introduced by Orloff [31] and was later proved to be NP-hard by Lenstra and Rinnooy Kan [27]. Golden and Wong [19] introduced an extension of the RPP, called the Capacitated Arc Routing Problem, in which demands are associated with the arcs and edges, and these must be optimally collected by a set of capacitated vehicles based at a depot.

Many different formulations and exact and heuristics algorithms have been proposed for the above-mentioned problems and their extensions, but they will be the subject of other chapters in this book. As a complement we refer the interested readers to the following surveys and books: Eiselt, Gendreau, and Laporte [14] and [15], Assad and Golden [3], Dror [10], Wøhlk [37], and Corberán and Prins [9].

1.5 ▪ Arc routing today

Over the past 35 years or so, arc routing has evolved into a mature research field. The initial impetus took place in the late 1970s and early 1980s under the scientific leadership of Christofides at Imperial College and of Bodin and Golden at the University of Maryland. Today, several research groups in various parts of the world are active in arc routing. Over the past years, major developments have occurred in the design of exact

Figure 1.16. *Attendants at WARP1: Juan P. Riquelme-Rodrígues, José M. Sanchis, Richard Eglese, Enrico Bartolini, Lukas Bach, André Langevin, Kevin Gaze, Gilbert Laporte, Ángel Corberán, Stefan Irnich, Bruce Golden, José M. Belenguer, Sanne Wøhlk, Manuel Sorge, Elisabeth Gussmagg-Pfliegl, Geir Hasle, Ana C. Nunes, Thais Ávila, Enrique Benavent, Leonor Pinto, Luc Muyldermans, Elena Fernández, Marcus Poggi, Stefan Ropke, Harilaos Psaraftis, and Jörg Kalcsics (Claudia Archetti, Alexander Butsch, Demetrio Laganá, and Vittorio Maniezzo are not in the photo).*

integer linear programming algorithms based on branch-and-cut, column generation, and their hybridization. In parallel, we have witnessed the development of powerful metaheuristics for arc routing problems, as well as the emergence of new complexity results. Several of the newer findings were presented at the 1st Workshop on Arc Routing Problems (WARP1) held in Copenhagen in May 2013 (see Figure 1.16).

1.6 ▪ Arc routing tomorrow

The development of exact and heuristic algorithms for arc routing problems has now reached a very high level of sophistication. This means that most classical variants described in this book can now be solved quite accurately for ever increasing sizes and within reasonable computation time limits. At the same time, the state-of-the-art methodologies are becoming increasingly difficult to reproduce and to teach. We may even have reached a point where it may be desirable to devise simpler algorithms, even if this is likely to result in a small loss of accuracy. In terms of applications, it is highly likely that new sophisticated variants of arc routing problems will emerge in the coming years, namely, in the areas of arc routing problems with profits, arc routing problems with synchronization components, and combinations of node and arc routing problems, as well as stochastic and dynamic problems. The latter class of problems is likely to become increasingly relevant in a context where vehicles are now often equipped with communication technologies which allow real-time interactions and decisions to be made.

Acknowledgments

Ángel Corberán wishes to thank the Ministerio of Economía y Competitividad (project MTM2012-36163-C06-02) of Spain and the Generalitat Valenciana (project PROMETEO/2013/049) for their support.

Bibliography

[1] D. Ahr, *Contributions to Multiple Postmen Problems*, Ph.D. thesis, University of Heidelberg, Heidelberg, Germany, 2004.

[2] A.A. Assad, *Leonhard Euler: A brief appreciation*, Networks, 49 (2007), pp. 190–198.

[3] A.A. Assad and B.L. Golden, *Arc routing methods and applications*, in Network Routing, M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhausser, eds., Handbooks of Operations Research and Management Science 8, North-Holland, Amsterdam, 2000, pp. 375–483.

[4] R. Bellman and K.L. Cooke, *The Königsberg bridges problem generalized*, Journal of Mathematical Analysis and Applications, 25 (1969), pp. 1–7.

[5] E. Benavent, Á. Corberán, and J.M. Sanchis, *Linear programming based methods for solving arc routing problems*, in Arc Routing: Theory, Solutions and Applications, M. Dror, ed., Kluwer, Boston, 2000, pp. 231–275.

[6] N.L. Biggs, E.K. Lloyd, and R.J. Wilson, *Graph Theory 1736–1936*, Clarendon Press, Oxford, UK, 1998.

[7] P. BRUCKER, *The Chinese postman problem for mixed graphs*, in Proceedings of the International Workshop, Bad Honnef, 1980, Lecture Notes in Computer Science 100, Springer, Berlin, 1981, pp. 354–366.

[8] N. CHRISTOFIDES, *The optimum traversal of a graph*, Omega, 1 (1973), pp. 719–732.

[9] Á. CORBERÁN AND C. PRINS, *Recent results on arc routing problems: An annotated bibliography*, Networks, 56 (2010), pp. 50–69.

[10] M. DROR, *Arc Routing : Theory, Solutions and Applications*, Kluwer, Boston, 2000.

[11] J. EDMONDS, *The Chinese postman's problem*, Operations Research Bulletin, 13 (1965), pp. B73–B77.

[12] ——, *Paths, trees and flowers*, Canadian Journal of Mathematics, 17 (1965), pp. 449–467.

[13] J. EDMONDS AND E.L. JOHNSON, *Matching, Euler tours and the Chinese postman problem*, Mathematical Programming, 5 (1973), pp. 88–124.

[14] H.A. EISELT, M. GENDREAU, AND G. LAPORTE, *Arc routing problems, part 1: The Chinese postman problem*, Operations Research, 43 (1995), pp. 231–242.

[15] ——, *Arc routing problems, part 2: The rural postman problem*, Operations Research, 43 (1995), pp. 399–414.

[16] L. EULER, *Solutio problematis ad geometriam situs pertinentis*, Commentarii Academiae Scientiarum Imperialis Petropolitanae, 8 (1736), pp. 128–140.

[17] L.R. FORD AND D.R. FULKERSON, *Flows in Networks*, Princeton University Press, Princeton, NJ, 1962.

[18] G.N. FREDERICKSON, *Approximation algorithms for some postman problems*, Journal of the Association for Computing Machinery, 26 (1979), pp. 538–554.

[19] B.L. GOLDEN AND R.T. WONG, *Capacitated arc routing problems*, Networks, 11 (1981), pp. 305–315.

[20] I. GRIBKOVSKAIA, Ø. HALSKAU, AND G. LAPORTE, *The bridges of Königsberg, a historical perspective*, Networks, 49 (2007), pp. 199–203.

[21] M. GRÖTSCHEL AND Y.-X. YUAN, *Euler, Mei-Ko Kwan, Königsberg, and a Chinese postman*, Documenta Mathematica, Extra Volume ISMP (2012), pp. 43–50.

[22] M.G. GUAN, *Graphic programming using odd or even points*, Chinese Mathematics, 1 (1962), pp. 237–277.

[23] M.G. GUAN, *On the windy postman problem*, Discrete Applied Mathematics, 9 (1984), pp. 41–46.

[24] C. HIERHOLZER, *Über die Möglichkeit, einen Linienzug ohne Wiederholung und ohne Unterbrechung zu umfahren*, Mathematische Annalen, 6 (1873), pp. 30–32.

[25] B. HOPKINS AND R.J. WILSON, *The truth about Königsberg*, The College Mathematics Journal, 35 (2004), pp. 198–207.

[26] D. KÖNIG, *Theorie der endlichen und unendlichen Graphen*, Akademische Verlagsgesellschaft, Leipzig, 1936.

[27] J.K. LENSTRA AND A.H.G. RINNOOY KAN, *On general routing problems*, Networks, 6 (1976), pp. 273–280.

[28] J.B. LISTING, *Vorstudien zur Topologie*, Vandenhoeck und Ruprecht, Göttingen, 1847.

[29] E. MINIEKA, *The Chinese postman problem for mixed networks*, Management Science, 25 (1979), pp. 643–648.

[30] Y. NOBERT AND J.-C. PICARD, *An optimal algorithm for the mixed Chinese postman problem*, Networks, 27 (1996), pp. 95–108.

[31] C.S. ORLOFF, *A fundamental problem in vehicle routing*, Networks, 4 (1974), pp. 35–64.

[32] C.H. PAPADIMITRIOU, *On the complexity of edge traversing*, Journal of the Association for Computing Machinery, 23 (1976), pp. 544–554.

[33] L. POINSOT, *Mémoire sur les polygones et les polyèdres*, Journal de l'École Polytechnique, 4 (1810), pp. 16–49.

[34] G. TARRY, *Le problème des labyrinthes*, Nouvelles Annales de Mathématiques, 14 (1895), pp. 187–190.

[35] O. VEBLEN, *An application of modular equations in analysis situs*, Annals of Mathematics, 2 (1913), pp. 86–94.

[36] R.J. WILSON AND J.J. WATKINS, *Graphs: An Introductory Approach*, Wiley, New York, 1990.

[37] S. WØHLK, *A decade of capacitated arc routing*, in The Vehicle Routing Problem: Latest Advances and New Challenges, B.L. Golden, S. Raghavan, and E.A. Wasil, eds., Springer, New York, 2008, pp. 29–48.

[38] F.J. ZARAGOZA, *Postman Problems on Mixed Graphs*, Ph.D. thesis, University of Waterloo, Waterloo, Ontario, Canada, 2003.

Chapter 2

The Complexity of Arc Routing Problems

René van Bevern
Rolf Niedermeier
Manuel Sorge
Mathias Weller

2.1 ▪ Introduction

This chapter is devoted to surveying aspects of computational complexity for three central arc routing problems (and their corresponding variants):

- CHINESE POSTMAN, where one asks for a minimum-cost tour traversing all edges of a graph at least once;
- RURAL POSTMAN, which generalizes CHINESE POSTMAN in the sense that *only a subset* of the edges has to be visited; and
- CAPACITATED ARC ROUTING, representing the most general arc routing problem in this chapter, allows more than one vehicle to be used to traverse the edges.

With the exception of the basic version of CHINESE POSTMAN, almost all of these problems are NP-hard; that is, they are computationally intractable problems with respect to the worst-case running times of existing solving algorithms (see Garey and Johnson [41]). Due to their practical importance, however, it is of interest to search for ways to cope with their computational intractability. In theoretical computer science—where strong emphasis is laid on *provable* performance bounds—there are basically two lines of attack:

- polynomial-time approximation algorithms [6, 92, 95], where one trades solution optimality for efficient, polynomial running time instead of exponential, as to be expected for finding exact solutions; and
- parameterized algorithmics and complexity analysis [24, 34, 75], where one seeks to identify small problem-specific parameters that influence the seemingly unavoidable exponential-time behavior. Then, algorithms for finding exact solutions in a running time that is exponential only in the parameter value can be designed.

Besides classical complexity classification along the lines of NP-hardness, we will discuss for all problems aspects of polynomial-time approximability as well as parameterized complexity. Notably, parameterized complexity classification was not subject of the previous surveys [25, 31, 30, 63, 83], and it is still a very young field in arc routing with numerous challenging research questions. Indeed, throughout the chapter we will feature several open questions targeted at stimulating future research.

The outline of our presentation is as follows. Different from the older survey by Dror [25], we follow a problem-wise organization. That is, for each of the described arc routing problems we discuss it and some of its interesting variants in terms of classical computational complexity, polynomial-time approximability, and parameterized complexity. As to the problems, we basically follow a strategy from easier to more demanding problems; that is, typically the later a problem comes, the harder (in computational terms) or more general it is. Before doing so, however, in the remainder of this introductory section we briefly describe the central concepts used throughout this chapter.

2.1.1 ▪ Eulerian graphs and notational conventions

We use standard notation for directed and undirected graphs. In particular, *graphs* are pairs (V,E) of a set V of *vertices* and a set E of *edges* (unordered pairs of vertices). Instead of a set of edges, directed graphs have a set A of ordered pairs of vertices called *arcs*. Mixed graphs contain both edges and arcs (a mixed graph is a triplet (V,E,A)). *Multigraphs* may contain multiple copies of each edge/arc, that is, the set of edges/arcs is a multiset. Let $G = (V,E)$ and let $u \in V$. A vertex $v \in V$ is said to be *adjacent* to u if $\{u,v\} \in E$. The number of vertices that are adjacent to a vertex u is denoted by $\deg_G(u)$ and called the *degree* of u in G. The vertex u is called *even* or *balanced* in G if $\deg_G(u)$ is even. A *path* from u to w in G is a sequence $(u = v_0, v_1, v_2, \ldots, v_\ell = w)$ of vertices in V such that $\{v_i, v_{i+1}\} \in E$ for all $0 \leq i < \ell$. Its *length* is ℓ, and it *traverses* or *visits* the edges $\{v_i, v_{i+1}\}$ for all $0 \leq i < \ell$. If such a path exists in G, then u is said to be *connected to* w. A *cycle* or *tour* is a path whose first and last vertices are identical. A subgraph G' of a graph G is *connected* if each two vertices of G' are connected in G'. If G' is maximal with respect to the number of edges, then it is called a *connected component* of G.

In directed graphs, slight variations of these concepts exist. For a vertex u its *incoming arcs* are all arcs of $A \cap (V \times \{u\})$ and its *indegree* is the number of such arcs in G. The *outgoing arcs* of u and the *outdegree* of u are defined analogously. We call u *balanced* if its indegree equals its outdegree.

The concept of paths and cycles trivially extends to directed graphs. A subgraph G' of a directed graph G is called *strongly connected* if each two vertices are connected by a directed path. It is *weakly connected* or simply *connected* if the undirected (multi)graph resulting from ignoring the arc directions is connected.

A *postman tour* in a graph is a tour visiting each edge of a certain set of edges at least once. An *Eulerian tour* in a graph is a tour visiting each edge exactly once. A graph that admits an Eulerian tour is called *Eulerian*. It is common knowledge that a graph is Eulerian if and only if it is connected and each vertex is balanced. This statement and the definitions of tours are analogous for directed graphs. A mixed graph $G = (V,E,A)$, however, is Eulerian if and only if the graph G' resulting from ignoring the directions of the arcs is connected, the degree of each vertex in G' is even, and for each subset $S \subseteq V$ of vertices, the number of arcs entering S minus the number of arcs leaving S is at most the number of edges between S and $V \setminus S$ (see Ford and Fulkerson [35]).

2.1.2 ▪ Classical complexity

In order to examine the computational complexity of a problem, we first have to define what a computational problem is. We will consider two types of problems:

A *decision problem* Q is a set of instances that fulfill some property. For example, the EVEN problem is the set of all even nonnegative integers. Equivalently, one can formulate the EVEN problem as "Given a number x, is x an even nonnegative integer?" In the context of this chapter, instances of decision problems are often pairs of a graph G and some number k. For example, the VERTEX COVER problem is the set of all pairs (G,k), such that the undirected graph G can be covered by at most k vertices. Herein, a graph is covered by a vertex set Z if all edges are incident to a vertex in Z. Equivalently we may define VERTEX COVER by the question "Given an undirected graph G and a number k, can G be covered by at most k vertices?"

An *optimization problem* Q is a function that, given some instance, outputs a feasible solution that minimizes (or maximizes) a measurement m_Q. For example, the *minimization variant* of the VERTEX COVER problem is "Given an undirected graph G, find a vertex cover Z of G such that $m_{\text{VERTEX COVER}}(Z) := |Z|$ is minimum." In the context of this chapter, the measurement m_Q is usually $m_Q(Z) = |Z|$ for vertex subsets Z of a graph or $m_Q(Z) = \sum_{e \in Z} c(e)$ for edge subsets Z of a graph with cost-function $c : E \to \mathbb{Q}$.

We say that a decision problem Q can be *solved* in some time $T(n)$ if there is an algorithm that, given an instance x of length $|x| = n$ bits, determines whether $x \in Q$ using at most $T(n)$ computation steps. Likewise, a minimization or maximization problem Q can be *solved* in time $T(n)$ if there is an algorithm that, given an instance x, computes a feasible solution Y for x in $T(n)$ computation steps such that $m_Q(Y)$ is minimum or maximum, respectively. Classical complexity theory investigates a zoo of interesting classes of decision problems (see Aaronson [2]), but we will consider only two of them here. The first class, called P, is the class of all decision problems that can be solved by a deterministic Turing machine in n^c time for some constant $c \in \mathbb{N}$. The second class, called NP, is the class of all decision problems that can be solved by a *non*deterministic Turing machine in n^c time for some constant $c \in \mathbb{N}$ [5, 36, 45, 79].

P vs. NP. The question of whether P = NP is both amazing and popular [36, 42, 45, 70, 79] since many problems faced by people in many sciences and businesses are easily seen to be in NP but no algorithm solving them "quickly," that is, in (deterministic) polynomial-time, is known.

Deciding whether or not P = NP is one of the seven famous Millennium Prize Problems stated by the Clay Mathematics Institute [1]: A solution to each of these problems is promised to be rewarded with a US$1,000,000 prize. The persistent inability to decide whether P = NP gave rise to the concept of polynomial-time reduction (see Arora and Barak [5]). We say that a decision problem A can be (polynomial-time) reduced to a decision problem B if there is some algorithm running in polynomial time and, given an instance x of A, produces an instance y of B such that $x \in A$ if and only if $y \in B$. This allowed the concept of so-called "hard problems" for the class NP: A problem Q is said to be NP-*hard* if all problems in NP can be reduced to it. If Q is also contained in NP, then it is called NP-*complete*. In the last decades, thousands of problems have been shown to be NP-hard (mostly by using the transitivity of reducibility that allows showing NP-hardness of a problem Q by reducing any NP-hard problem to Q). A large subset of these

problems has been listed by Garey and Johnson [41]. Since it is widely believed, we will assume $P \neq NP$ in the remainder of this chapter.

Coping with NP-hardness. Many real-world problems have been shown to be NP-hard and, thus, do not admit polynomial-time algorithms. Since we can use algorithms that solve optimization problems to solve decision problems, this also implies that there are no polynomial-time algorithms to minimization or maximization problems corresponding to NP-hard decision problems. Since we cannot simply refuse to solve these problems, ways of coping with this computational hardness have been developed. One possibility is to accept a feasible, yet not necessarily optimal, solution that can be shown to be reasonably close to the optimum. Many problems have been shown to admit such *approximation algorithms* [6, 92, 95]. Another approach to coping with NP-hard problems exploits the fact that the instances produced by reductions are pathological for most applications; that is, aspects that make the considered problem hard are often unlikely to be seen in practice. Thus, it might be possible to design an algorithm that is fast for instances that are practically relevant but takes exponential time for other instances. A theoretical approach incorporating this thought is *Parameterized Complexity* [24, 34, 75]. Another theoretical approach in this direction is the (relatively new) technique of *Smoothed Analysis* introduced by Spielman and Teng [89, 90] that diverges from the concept of worst-case analysis, which "is often the source of discrepancy between the theoretical evaluation of an algorithm and its practical performance" [90]. Here, the performance of algorithms is measured as expected running time when input instances are exposed to a slight random perturbation, thus eliminating the influence of pathological worst-case instances. Unfortunately, there seem to be no works considering routing problems under smoothed analysis yet.

2.1.3 ▪ Complexity of approximation

As mentioned, one approach of coping with NP-hardness is to give up the search for an optimal solution and accept a slightly less accurate answer [6, 92, 95]. For instance, VERTEX COVER cannot be solved in polynomial time, but we can approximate the answer to the minimization variant in polynomial time. More precisely, with OPT_G denoting the optimum solution of the minimization variant of VERTEX COVER on input G, a feasible solution (that is, a vertex cover) Z for G such that $1 \leq |Z|/|\mathrm{OPT}_G| \leq 2$ can be computed in polynomial time: Find an edge $\{u, v\}$ in G such that neither u nor v are selected, select both u and v, and repeat until no such edge can be found.

In general, for a minimization problem Q, we define $\mathrm{OPT}(x)$ as the optimal solution for an instance x. Then, a polynomial-time algorithm that, given an instance x of Q, computes a feasible solution Z such that $1 \leq m_Q(Z)/m_Q(\mathrm{OPT}(x)) \leq d(|x|)$ is called a *factor $d(|x|)$ approximation* for Q. Likewise, for a maximization problem Q, a factor $d(|x|)$ approximation algorithm computes a feasible solution Z with $1 \leq m_Q(\mathrm{OPT}(x))/m_Q(Z) \leq d(|x|)$. Most commonly, the function d is constant and, thus, factor d approximations are called *constant-factor* approximations. However, some problems are known not to admit constant-factor approximations [6, 92, 95].

2.1.4 ▪ Parameterized complexity

Parameterized algorithmics [24, 34, 75] is an approach to find optimal solutions for NP-hard problems in reasonable time. The idea is to accept the seemingly inevitable combinatorial explosion, but to confine it to one aspect of the problem, the *parameter*.

Consider a decision problem Q. If there is an algorithm A such that, for all $p \in \mathbb{N}$ and all instances x of Q whose parameter is p, A decides whether $x \in Q$ in polynomial time, then we say Q is polynomial-time solvable for constant parameter values. The class of all such problems is called XP. Algorithms for problems in XP run in time $n^{f(p)}$ where p is the parameter and f is some computable function. For practical applications, however, such algorithms often still take too much time. Thus, the following refinement rose: A decision problem Q is called *fixed-parameter tractable* (FPT) with respect to a parameter p if there is an algorithm solving any instance x of Q in $f(k) \cdot |x|^c$ time for some computable function f and a constant c. For example, choosing the parameter "size k of the desired vertex cover" for VERTEX COVER, Chen, Kanj, and Xia [18] presented an algorithm running in $O(1.2738^k + k|V|)$ time, which can solve instances with small desired solution sizes fast. Note that the crucial difference between XP and FPT is that, while problems in both classes are solvable in polynomial time for constant parameter values, in the case of XP the degree of the corresponding polynomial may depend on the parameter, while in the case of FPT it may not.

Like in classical complexity, there is a concept of hardness in parameterized complexity. A class of problems that are conjectured to not be FPT is $W[1]$. A decision problem Q is called $W[1]$-hard with respect to the parameter p if Q being FPT with respect to p implies that all problems in $W[1]$ are also FPT. Similarly to the concept of NP-hardness, this can be achieved by a reduction concept [24, 34, 75].

Often, finding a good parameter that captures aspects that make a problem hard is not an easy task [32, 60, 76]. Multiple techniques exist to support the parameter search. For example, "deconstructing intractability" is the technique of analyzing the characteristics of proofs of computational worst-case hardness—proofs that often exploit parameter values unlikely to occur in practice. Other techniques include "distance from triviality" and "above guarantee" parameterizations [17, 48, 61, 68, 76].

Problem kernelization. An important property of parameterized design and analysis of algorithms is that it provides means of measuring the effectiveness of preprocessing. Consider, for example, some algorithm that, given an instance (G, k) of VERTEX COVER, produces a simpler instance of VERTEX COVER in polynomial time. In classical complexity, simpler means smaller. Then, however, if such an algorithm existed, we could solve VERTEX COVER in polynomial time by iterating the described algorithm. By introducing the parameter k, however, an algorithm is possible that produces smaller instances until some threshold measured in k is reached. For example, the books of Downey and Fellows [24], Flum and Grohe [34], and Niedermeier [75] present polynomial-time algorithms that, given an instance (x, k) of VERTEX COVER, produce an equivalent instance (x', k') of VERTEX COVER with at most $2k'$ vertices. In general, a *kernelization* (or *problem kernel*) of a decision problem Q is a polynomial-time algorithm that, given an instance x with parameter p of Q, computes an instance x' with parameter p' of Q such that, for some functions $f, g : \mathbb{N} \to \mathbb{N}$,

- $x \in Q$ if and only if $x' \in Q$,
- $|x'| \leq g(p)$, and
- $p' \leq f(p)$.

The function g is then called the *size* of the problem kernel and constitutes a measure of effectiveness of the kernelization algorithm. A common method to develop such preprocessing algorithms is to provide polynomial-time executable *data reduction rules* [14, 49, 66]. Kernelization is a core tool of parameterized algorithmics and perhaps the concept in parameterized algorithmics with the widest practical relevance (see Guo and Niedermeier [49] and Hüffner, Niedermeier, and Wernicke [56]). In the recent past, Bodlaen-

Table 2.1. *Complexity of the* CHINESE POSTMAN (CP) *problem.*

CP variant	Classical complexity
Undirected	$O(\|V\|^3)$-time algorithm [29]
Directed	$O(\|V\|^3)$-time algorithm [29]
	$O((\|E\|-\|V\|)\|V\|^2)$-time algorithm [65]
Mixed	NP-complete [78]
	$O(\|V\|^3)$-time solvable if each vertex has even degree [29]
Windy	NP-complete [47]
	in P in some special cases [47, 97, 101]
k-Hierarchical	NP-complete [27]
	$O(k\|V\|^4)$-time solvable if precedence relation linear [62]
Min-sum k postmen	$O(\|V\|^3)$-time algorithm with post office vertex [80, 102], otherwise NP-complete [91, 52]
Min-max k postmen	NP-complete [40]
CP variant	**Approximation**
Mixed	$O(\max\{\|V\|^3, \|A\|(\max\{\|A\|, \|E\|\})^2\})$-time factor 3/2 [85]
Windy	factor 3/2 [84]
Min-max k postmen	$O(\|V\|^3)$-time factor $(2-1/k)$ [40]
CP variant	**Parameterized complexity**
Mixed	$O(2^{\|E\|} \cdot \|V\|^3)$-time algorithm [50]
	in FPT with respect to $\|A\|$ [50]
	in XP with respect to treewidth [33]
Min-sum k postmen	in FPT with respect to k without post office vertex [51, 52]

der et al. [15] and Fortnow and Santhanam [37] developed methods to show that certain problems do not admit polynomial-size kernels. Most of these techniques build on the hypothesis that the polynomial hierarchy does not collapse to the second level (which implies that P $\neq$ NP).

2.2 ▪ The Chinese Postman Problem

The CHINESE POSTMAN problem is to find a minimum cost tour in a graph with edge costs such that each edge is traversed at least once. As CHINESE POSTMAN is one of the most central arc routing problems, there are many interesting studies from a complexity-theoretic perspective. In its original form, this problem is solvable in polynomial time. As usual with problems motivated by real-world applications, there are many variants and possible side constraints. However, even side constraints that seem innocuous make CHINESE POSTMAN NP-hard. Here, we review the computational complexity of CHINESE POSTMAN and some of its variants. As mentioned above, NP-hardness is not a reason for despair but rather stirs further investigation. Thus, we try to explain the combinatorial structure in the problems that is used to show hardness and reflect on the impact on practice. For an overview of the results that are presented in this section, see Table 2.1.

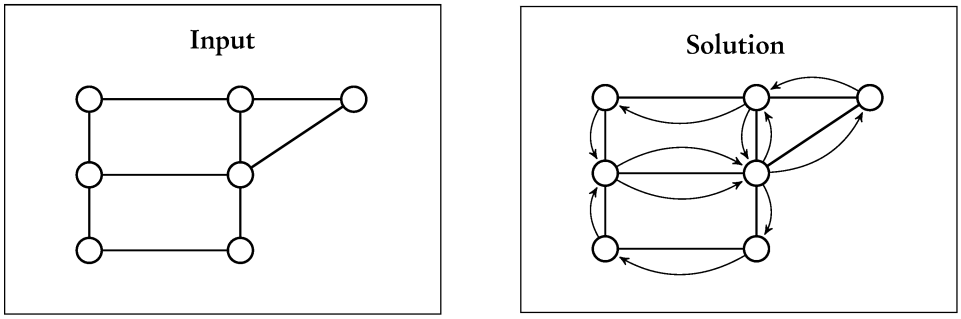


Figure 2.1. *Instance and solution for the* CHINESE POSTMAN *problem.*

2.2.1 ▪ The classical Chinese Postman Problem

Edmonds [28] and Guan [46] originally introduced CHINESE POSTMAN as the following problem:

CHINESE POSTMAN
Input: A connected, undirected graph $G = (V,E)$ with edge cost $c(e) \geq 0$ for every $e \in E$ and a maximum cost $c_{\max}$.
Question: Is there a tour that traverses every edge in E at least once and has cost at most $c_{\max}$?

Figure 2.1 depicts on the left an input graph with edge costs proportional to the length in the drawing and a minimum cost solution on the right.

Tractability. Edmonds and Johnson [29] showed that CHINESE POSTMAN is solvable in polynomial time. A strategy to efficiently decide CHINESE POSTMAN is based on the observation that any desired tour induces an Eulerian supergraph of G by multiplying every edge of G according to the number of times it is traversed by the tour. Obviously, the obtained supergraph is a multigraph. In the reverse direction, every such Eulerian supergraph also induces a postman tour of the same cost. CHINESE POSTMAN thus reduces to the problem of finding an Eulerian supergraph G' such that the cost of the edges in G' does not exceed $c_{\max}$.

Finding such an Eulerian supergraph can be done as follows. Recall that a graph is Eulerian if and only if it is connected and each vertex has even degree. By prerequisite, the input graph G is already connected, and it remains only to add edges in order to give every vertex even degree. It can be shown that this can be achieved by solving an equivalent minimum-cost perfect matching problem in an auxiliary graph G^* with edge costs: The graph G^* is complete and contains only the odd-degree vertices of G. Every edge is priced according to a shortest path between its endpoints in the graph G. The basic ideas for proving the equivalence are as follows. First, show that the excess edges in a minimum-cost Eulerian supergraph of G form paths between odd-degree vertices. Second, show that shortest paths according to two matching edges in G^* do not share any edge. Then one can easily transfer a matching in G^* to an Eulerian supergraph of G and vice versa. To construct the graph G^*, one can employ an all-pairs shortest path algorithm using $O(|V|^3)$ time and then solve the corresponding matching problem on G^* in $O(\sqrt{|V|}\cdot|E|)$ time (see Micali and Vazirani [73] for the matching running time). Thus, CHINESE POSTMAN is solvable in $O(|V|^3)$ time, and, therefore, it is in P.

Note that, in the above described way, one does not actually find a postman tour of the desired weight but ensures that one exists in the Eulerian supergraph. This suffices to decide CHINESE POSTMAN. Nevertheless, finding a tour if one exists can be done in linear time using Hierholzer's algorithm, which is described, for example, by Berge [12].

Directed graphs. If the input graph for CHINESE POSTMAN is directed, then one can proceed similarly as above. The goal is again to augment the input graph to an Eulerian graph using a minimum-cost set of arcs. A directed graph is Eulerian if and only if it is strongly connected and each of its vertices has an equal number of incoming and outgoing arcs. It can be shown that mending vertices for which this is not the case can be reduced to a minimum-cost network flow problem (see Edmonds and Johnson [29]) or solved using linear programs that have integral optimal solutions (see Beltrami and Bodin [8] and Christofides [19]). This yields running times of $O(|V|^3)$ and $O(|E| \cdot |V|^2)$, respectively. Lin and Zhao [65] later gave an alternative algorithm based on a network flow formulation that has $O((|E|-|V|) \cdot |V|^2)$ running time.

In practice, the plain CHINESE POSTMAN problem is often augmented with additional constraints or amended to better reflect its application. Below we consider the complexity of some of these variants. As we will see, it is easy to step from polynomial-time solvable into NP-hard terrain. Nevertheless, for practical scenarios there often are ways that allow one to preserve *efficient* solvability to some extent.

2.2.2 ▪ Mixed graphs

In practical instances it can happen that both (undirected) edges and (directed) arcs occur. For example, in snow plowing, it can happen that some streets are narrow enough such that they can be cleaned by traversing them only once in either direction, but other streets have to be plowed in both directions. In this case, we have to solve the MIXED CHINESE POSTMAN problem, as defined by Edmonds and Johnson [29]:

MIXED CHINESE POSTMAN
Input: A strongly connected, mixed graph $G = (V, E, A)$ with cost $c(e) \geq 0$ for every $e \in E \cup A$ and a maximum cost $c_{\max}$.
Question: Is there a (directed) tour that traverses every edge in E and every arc in A at least once and has cost at most $c_{\max}$?

If we assume P $\neq$ NP, then MIXED CHINESE POSTMAN is not polynomial-time solvable as we will see below. Nevertheless, there are some tractable special cases.

Note that the question of whether a given mixed graph admits a postman tour of any cost is easy to decide: One simply has to check whether the graph is strongly connected. This changes drastically if we restrict the postman tour to traverse each arc exactly once or if we restrict it to traverse each edge exactly once. As proved by Zaragoza Martínez [98, 99, 100], it is NP-hard to decide whether such tours exist. This result has some applications to hardness of approximability, which we will touch on below.

2.2.2.1 ▪ Hardness

MIXED CHINESE POSTMAN has been shown NP-complete by Papadimitriou [78]. The main difficulty in solving MIXED CHINESE POSTMAN lies in choosing orientations for the (undirected) edges when we are given a tight budget for our tour and can only afford to traverse each edge once. We then have to orient the edges and add some further arcs in

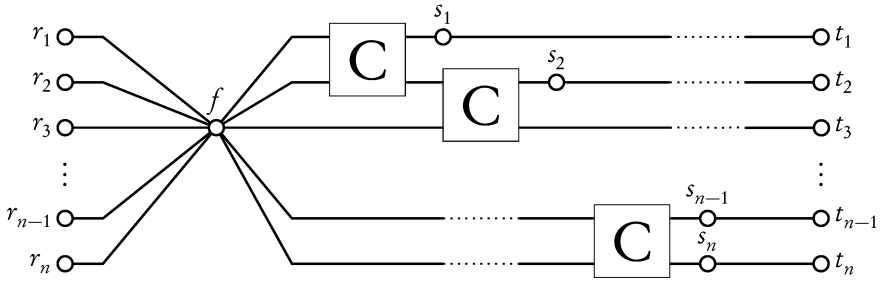


Figure 2.2. *Variable gadget for reducing* 3SAT *to* MIXED CHINESE POSTMAN.

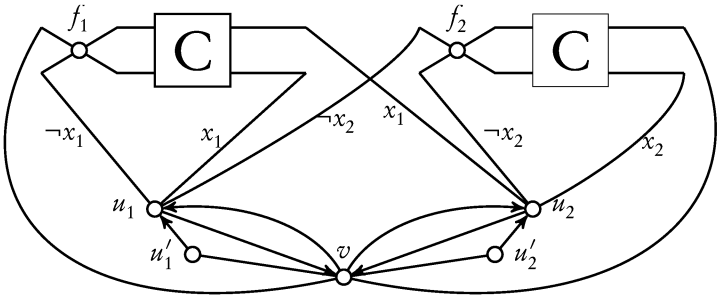


Figure 2.3. *Example for NP-hardness reduction of* MIXED CHINESE POSTMAN.

order to obtain a directed Eulerian graph, that is, to make every vertex balanced. If there are multiple edges incident to one vertex, it is not an easy task to determine the correct orientation of each edge.

In fact, one can exploit this difficulty to reduce 3SAT to MIXED CHINESE POSTMAN; we sketch some ideas of the reduction here. Suppose we can construct a mixed graph with edge costs as shown in Figure 2.2. This graph will represent a variable of the instance of 3SAT. The boxes labeled C represent graphs with the following property. They can be made Eulerian—up to the vertices that connect to the rest of the graph—by adding arcs of cost at most two only in two ways: by orienting every outgoing edge to the right or to the left (see Papadimitriou [78] for details). Hence, the variable gadget of Figure 2.2 has the property that, if we are given a budget of at most $2(n-1)$, we can orient all of its edges either to the right or to the left. We can use this property to represent the two possible truth values via the orientations.

The main idea for representing clauses is shown via an example in Figure 2.3 (reproduced from Papadimitriou [78]). It shows an instance of MIXED CHINESE POSTMAN constructed from the trivial 3SAT formula $(\neg x_1 \vee x_1 \vee \neg x_2) \wedge (x_1 \vee \neg x_2 \vee x_2)$. At the top, there are two variable gadgets corresponding to x_1 and x_2. The two clauses are represented by the pairs of vertices u_1, u_1' and u_2, u_2' and their connections to v and the variable gadgets. The vertex v is a dummy vertex that takes surplus connections from the variable gadgets which are not needed to represent them in clauses.

Consider the vertex u_1 and orientations of its incident edges. If the budget for our tour and the costs of the incoming arcs of u_1 are chosen such that we can traverse each of them only once, then, at least one of the edges incident to u_1 has to be oriented toward it. Otherwise, we could not balance u_1. By the way the edges are connected to u_1 and to the variable gadgets, the orientation corresponds to satisfying the clause that u_1 represents.

Papadimitriou [78] also showed that the reduction sketched above can be modified to yield NP-hardness even for very restricted instances as stated in the following theorem.

Theorem 2.1 (Papadimitriou [78]). MIXED CHINESE POSTMAN *is* NP*-complete, even if the input graph is planar, each vertex has degree at most three, and each edge and arc has cost one.*

2.2.2.2 ▪ Tractability

Despite the above hardness result even for very restricted instances, there are some tractable special cases of MIXED CHINESE POSTMAN.

Even degree. If every vertex has an even number of incident edges and arcs, Edmonds and Johnson [29] showed that MIXED CHINESE POSTMAN is polynomial-time solvable by reduction to a network flow problem. This special case is interesting since when modeling street networks, it is expected that many vertices obtain exactly four edges or arcs. Indeed, the hardness reductions given by Papadimitriou [78] and Zaragoza Martínez [98, 99, 100] all construct instances with an unbounded number of odd-degree vertices. Thus, in the spirit of "deconstructing intractability" (see Komusiewicz, Niedermeier, and Uhlmann [61] and Niedermeier [76]), it would also be interesting to know the complexity of MIXED CHINESE POSTMAN when there are only a few vertices with an odd number of incident edges or arcs. To the best of our knowledge, this is currently an open question.

Challenge 1. *Devise an algorithm for* MIXED CHINESE POSTMAN *that witnesses fixed-parameter tractability with respect to the parameter "number of odd-degree vertices," or show that such an algorithm is unlikely to exist.*

Few edges or arcs. When the number of (undirected) edges in the input graph is small, MIXED CHINESE POSTMAN can be solved efficiently. This is because, in an optimum postman tour, every edge is traversed either forward, backward, or in both directions. Observe that, once we know which one is true for every edge, we can replace the edges by (directed) arcs and solve the resulting instance of directed CHINESE POSTMAN in polynomial time. Thus, guessing one of the three options for every edge, we obtain an algorithm running in $O(3^{|E|} \cdot |V|^3)$ time. Gutin, Jones, and Sheng [50] showed how this running time can be improved to $O(2^{|E|} \cdot |V|^3)$ using network flows. It is reasonable to assume that $|E|$ is small in some practical applications, for example, in snow plowing, where it is rare that a street can be cleaned by traversing it only once. However, the reverse is true for some other applications, for example, in police patrols, where the direction of traversal does not matter except for one-way streets. If the number of one-way streets is small, then an FPT algorithm with respect to the parameter number $|A|$ of arcs might be handy. Indeed, Gutin, Jones, and Sheng [50] described such an algorithm. The procedure includes an application of the "treewidth reduction" technique [71] and dynamic programming, iteratively computing partial solutions until a complete one is obtained. As a consequence of the generality of treewidth reduction, the number of partial solutions one has to keep track of in the dynamic program can be rather large, of order $\Omega(2^{2^{|A|}})$. FPT algorithms are widely regarded as promising for practice if their running time is "single exponential," that is, $O(c^k \cdot n^d)$ for some constants c, d, parameter k, and input size n. Hence, an interesting open question is whether Gutin, Jones, and Sheng's algorithm can be improved to such a running time, at least on the practically relevant planar graphs.

Challenge 2. *Give an algorithm for planar* MIXED CHINESE POSTMAN *with* $O(c^{|A|} \cdot |V|^d)$ *running time for constants* c, d*; or show that such an algorithm is unlikely to exist.*[1]

Small treewidth. Furthermore, Fernandes, Lee, and Wakabayashi [33] proved that MIXED CHINESE POSTMAN is in XP with respect to the parameter "treewidth of the input graph." That is, it can be solved in polynomial time if the treewidth is constant. Intuitively, treewidth measures the tree-likeness of graphs. For example, trees have treewidth one, and cycles have treewidth two (see Bodlaender and Koster [16] for a recent survey on treewidth). However, the exponent in the polynomial of the running time depends exponentially on the treewidth. Thus, this result is of mainly theoretical interest. It also raises the question whether MIXED CHINESE POSTMAN is FPT with respect to the treewidth.

Further tractable cases. There are also two polynomial-time solvable special cases for MIXED CHINESE POSTMAN that can be obtained by transforming it into WINDY CHINESE POSTMAN, which we will consider in Subsection 2.2.3.

2.2.2.3 ▪ Approximability

Motivated by the NP-hardness of MIXED CHINESE POSTMAN, the question was pursued of how close one can get to an optimal solution using only polynomial time. This was successful in that Raghavachari and Veerasamy [85] developed an algorithm that approximates the cost of the optimal postman tour within a factor of 3/2, improving the previously known factor 3/2 approximation algorithm, given by Frederickson [39], that was limited to planar graphs. However, Zaragoza Martínez [98, 99, 100] proved that it is not possible to polynomial-time-approximate the cost of *deadheading* within any constant factor, where deadheading means traversing arcs or edges more than once.

A factor 2 approximation. The origin of the currently best known approximation algorithm for MIXED CHINESE POSTMAN is a heuristic proposed by Edmonds and Johnson [29]. It is based on the following sufficient condition for the existence of an Eulerian tour: If in a mixed graph each vertex has even degree—a vertex in a mixed graph has even degree if it has an even number of incident arcs and edges—and each vertex has an equal number of incoming and outgoing arcs, then the graph has an Eulerian tour, as noted by Ford and Fulkerson [35]. Edmonds and Johnson's heuristic first produces even degree by ignoring the direction of the arcs and using a matching method analogous to that used for solving the undirected CHINESE POSTMAN. Using a network flow formulation, each vertex in the resulting graph is then given an equal number of incoming and outgoing arcs in a cost-optimal way. Frederickson [39] showed that the second step can be carried out such that it preserves even degrees. In this way, he obtained a factor 2 approximation algorithm for the tour cost. Roughly, the argument to show this is as follows: Let us duplicate each edge and arc in the input graph G to obtain a multigraph G_2. In G_2, each vertex has even degree, and the above algorithm will solve the resulting instance optimally. Duplicating an optimum solution for G yields a feasible solution for G_2, and, hence, we have that the algorithm gives a solution for G_2 of cost at most twice the optimum of G. Then, one can show that the edges added to G due to the matching are a subset of the duplicated edges in G_2 and, furthermore, that producing equal in- and outdegrees for each vertex after the matching is at most as costly in G as in G_2. Thus, we obtain that the algorithm

[1] In an earlier version of this chapter, the authors asked whether MIXED CHINESE POSTMAN is FPT with respect to $|A|$. This question was also featured in a survey talk by Sorge [86]. Subsequently, Gutin, Jones, and Sheng [50] resolved the challenge, showing fixed-parameter tractability.

produces a solution of cost at most the one in G_2, which is at most twice the optimum for G.

A mixed strategy for factor 5/3. Frederickson [39] showed that a "reverse" algorithm also gives a factor 2 approximation, that is, an algorithm that first gives every vertex equal number of incoming and outgoing arcs and then produces even degree for every vertex while preserving the first property. He observed that the two algorithms perform optimally on each others' worst-case examples, and, in fact, he analyzed their solution cost to be at most $C^* + 2C_{\leftrightarrows}$ and $2C^* - C_{\leftrightarrows}$ for the reverse algorithm. Here, C^* is the cost of the optimal postman tour and $C_{\leftrightarrows}$ is the cost of the step that produces equal in- and out-degree. Thus, the solution cost guarantees coincide when $C_{\leftrightarrows} = C^*/3$, and, hence, one of the guarantees is always at most $5C^*/3$, giving a factor $5/3$ approximation algorithm when choosing the best solution.

Improving the mixed strategy to factor 3/2. The approximation factor of the above algorithm has later been improved by Raghavachari and Veerasamy [85]. They adopted the algorithms as presented by Frederickson [39] and made a modification to get a solution cost guarantee of $C^* + C_{\leftrightarrows}$ instead of $C^* + 2C_{\leftrightarrows}$ for the first of the two algorithms. Now the guarantees for the solution cost coincide when $C_{\leftrightarrows} = C^*/2$, and, hence, one gets a factor $3/2$ approximation algorithm. The crucial observation for the better bound is that a lower bound on the optimal solution used by Frederickson can be tightened to also include $C_{\leftrightarrows}$, which then "swallows" one of the $C_{\leftrightarrows}$ summands in the solution cost bound above.

Theorem 2.2 (Raghavachari and Veerasamy [85]). MIXED CHINESE POSTMAN *can be factor* $3/2$ *approximated in* $O(\max\{|V|^3, |A|(\max\{|A|, |E|\})^2\})$ *time.*

Inapproximability of the deadheading cost. Since a postman tour always includes the edges and arcs of the input graph, one might wonder whether one can approximate the cost of additional edge and arc traversals, also called *deadheading* cost, rather than the whole tour cost. However, Zaragoza Martínez [98, 100] proved that this is not possible within any constant factor and in polynomial time. He obtained this result by proving NP-hardness of deciding whether a given mixed graph has a postman tour that traverses each edge exactly once. The proof is by reduction from the NP-hard NOT ALL EQUAL SATISFIABILITY problem. Given this hardness result, consider a mixed graph $G = (V, E, A)$ and set the cost $c_a = 0$ for all $a \in A$ and the cost $c_e = 1$ for all $e \in E$. Then, G admits a postman tour that traverses each edge at most once if and only if it has a postman tour with zero deadheading cost. Thus, any constant-factor approximation algorithm could decide whether such a tour exists and, thus, cannot run in polynomial time.

2.2.3 ▪ Windy costs

Another natural variant of the CHINESE POSTMAN problem arises when considering asymmetric costs; that is, it may be harder to traverse a street in one direction than it is in the reverse one, modeling roads on a slope, or blowing wind. Minieka [74] formulated this as the following problem:

WINDY CHINESE POSTMAN
Input: A connected, undirected graph $G = (V,E)$ with two cost values $c(u,v)$, $c(v,u) \geq 0$ for every edge $\{u,v\} \in E$ and a maximum cost $c_{\max}$.
Question: Is there a (directed) tour of cost at most $c_{\max}$ that traverses every edge in E at least once?

Hardness. Unfortunately, this problem is NP-hard, as Guan [47] showed: MIXED CHINESE POSTMAN can be transformed into WINDY CHINESE POSTMAN in polynomial time. The idea for this transformation is to keep each (undirected) edge and to replace each pair of arcs $(u,v),(v,u)$ by an edge $\{u,v\}$ with the appropriate costs. Furthermore, replace each arc (u,v), for which (v,u) is not present, by an edge $\{u,v\}$ with cost $c(u,v)$ according to the original cost of (u,v) and very high cost $c(v,u) > c_{\max}$. Hence, the arc (v,u) cannot be traversed by any tour of cost at most $c_{\max}$. This gives a one-to-one correspondence between optimal solutions, and Theorem 2.1 implies that WINDY CHINESE POSTMAN is NP-hard even on planar graphs with maximum degree three.

Tractability. On the positive side, WINDY CHINESE POSTMAN is solvable in polynomial time if the cost of traversing each cycle in the input graph is the same regardless of the direction of traversal (see Guan [47]). WINDY CHINESE POSTMAN is also polynomial-time solvable if the input graph is Eulerian (see Win [97]).

Furthermore, Win [96, 97] started the line of research of classifying so-called *windy postman perfect* graphs. These are graphs for which a certain linear programming formulation yields integer solutions and, consequently, for which the corresponding integer linear program can be solved in polynomial time. In this regard, Zaragoza Martínez [101] more recently proved that, in particular, series-parallel graphs are windy postman perfect.

Approximability. Exploiting the polynomial-time solvability of WINDY CHINESE POSTMAN on Eulerian graphs, Win [97] obtained a factor 2 approximation algorithm. A simplified version of his idea works as follows: First, double all edges in the input graph, which is now Eulerian. Then, solve the resulting instance optimally. Since following the optimal tour for the input graph twice gives a feasible solution for the graph with the doubled edges, this gives a factor 2 approximation. Raghavachari and Veerasamy [84] later gave a factor 3/2 approximation algorithm by combining a mixed strategy similar to the one used for the approximation algorithm for MIXED CHINESE POSTMAN and a modification of Win's linear programming formulation for WINDY CHINESE POSTMAN.

2.2.4 ▪ Edge hierarchies and precedence relations

In practical applications, one sometimes has to solve a variant of CHINESE POSTMAN where some edges have to be traversed before other edges. For example, in snow plowing, main streets should be cleared before secondary streets and, in turn, secondary streets should be cleared before residential streets. Dror, Stern, and Trudeau [27] formalized this in the following problem. Let $G = (V,E)$ be an undirected graph, and let $P = \{E_1,\ldots,E_k\}$ be an edge partition, that is, $\bigcup_{i=1}^{k} E_i = E$ and $E_i \cap E_j = \emptyset$ for all $1 \leq i,j \leq k$. Call the sets E_i *priority classes*. Furthermore, let $R \subseteq P \times P$ be a partial ordering. We say that a tour in G *respects* R if for every $(E_i,E_j) \in R$ each edge in E_i is traversed before any edge in E_j is traversed.

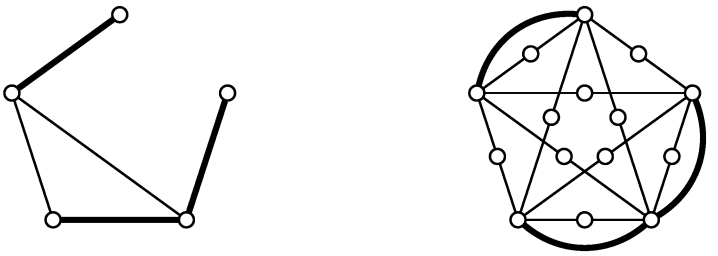


Figure 2.4. *Example for NP-hardness of* HIERARCHICAL CHINESE POSTMAN *even with two priority classes.*

HIERARCHICAL CHINESE POSTMAN
Input: A connected (undirected) graph $G = (V,E)$ with edge cost $c(e) \geq 0$ for every $e \in E$ and a maximum cost $c_{\max}$. Additionally, an edge-partition $P = \{E_1,\ldots,E_k\}$ and a partial ordering relation $R \subseteq P \times P$.
Question: Is there a tour that traverses every edge in E at least once, has cost at most $c_{\max}$, and respects R?

As we will see, this problem is NP-hard in general. However, there are three factors that seem to influence its complexity. Namely, whether the priority classes are connected; if so, their number; and whether the precedence relation is a linear ordering.

We note that a different HIERARCHICAL CHINESE POSTMAN formulation has been proposed by Alfa and Liu [4] for which, to our knowledge, there are no complexity-theoretic results yet.

Hardness with two priority classes. HIERARCHICAL CHINESE POSTMAN is NP-hard even when there are just two priority classes. This follows from observations by Dror, Stern, and Trudeau [27] via a reduction from the NP-hard RURAL POSTMAN problem, which we will consider more closely in Section 2.3. In UNDIRECTED RURAL POSTMAN one is given an undirected Graph $G = (V,E)$ with costs c_e for each $e \in E$ and a set of required edges $R \subseteq E$. The task is to find a minimum-cost tour that traverses each edge in R at least once.

To reduce RURAL POSTMAN to HIERARCHICAL CHINESE POSTMAN, first create a complete graph G' on the vertices V. Price each edge $\{u,v\}$ according to a shortest path between the vertices u and v in G. Then, replace each edge by a path with two edges and split the original cost equally between both new edges. Place all edges constructed so far in the priority class E'_1. This is the first step. In the second and final step, simply add all edges in R to the constructed graph G' and price them according to their original cost. The edge partition is $\{E'_1,R\}$, and the precedence relation is defined by $E'_1 < R$. An example of the reduction is shown in Figure 2.4 with a RURAL POSTMAN instance to the left, where bold lines represent required edges, and the corresponding HIERARCHICAL CHINESE POSTMAN instance on the right. There, thin lines represent edges in one priority class, and bold lines represent edges in a second priority class.

We claim that every rural postman tour in G with cost C implies a hierarchical Chinese postman tour in G' with cost $C + C_{E'_1}$ and vice versa. Here, $C_{E'_1}$ is the cost of the edges in E'_1. Without loss of generality, assume that $|V|$ is odd. Then, the graph constructed in the first step is Eulerian. Thus, given a rural postman tour T in G of cost C, we obtain a hierarchical Chinese postman tour in G' of cost $C + C_{E'_1}$ by first traversing all edges in E'_1 and then tracing T in G', following edges in R whenever possible and pairs of

edges corresponding to shortest paths, otherwise. Conversely, every hierarchical Chinese postman tour has to first traverse each edge in E'_1. It is not hard to see that the tour can be partitioned into a tour for E'_1 and a tour corresponding to a rural postman tour for G: Consider a hierarchical Chinese postman tour T for G' and the Eulerian multigraph M_T it induces by multiplying each edge according to how often it is traversed. Remove the edge set E'_1 from M_T to obtain M'_T. Up to isolated vertices, the graph M'_T is still Eulerian: Since $|V|$ is odd, removing E'_1 removes an even number of edges for each vertex, resulting in even degrees. Connectedness of M'_T follows, because for two vertices that are incident to required edges, there is a path between them in T *after* the initial traversal of E'_1. Thus, T can be partitioned into a tour for E'_1 and the tour implied by M'_T.

Hardness with connected priority classes. The above considerations express the intuition that one has to solve RURAL POSTMAN as a subproblem if one of the priority classes is not connected. However, as Dror, Stern, and Trudeau [27] proved, HIERARCHICAL CHINESE POSTMAN remains NP-hard even if we demand that the priority classes be connected. Using their ideas, we modify the above sketch of a hardness proof to get hardness for connected priority classes: Instead of putting all required edges in R into one priority class, we have a priority class R_i for each connected component induced by the required edges. The new precedence relation is defined by $E'_1 < R_i$ for all R_i. It is still the case that each edge in E'_1 has to be traversed before any required edge. But the new precedence relation does not restrict in which order a tour traverses the required edges. Thus, the correctness follows as above.

Influence of the number of priority classes on complexity. Note that if the priority classes are connected, then it is necessary to have many priority classes in the above reduction: The standard hardness reduction for RURAL POSTMAN is from HAMILTONIAN CYCLE and creates one connected component in the required edges for each vertex in the original instance (see Section 2.3). This raises the question of whether HIERARCHICAL CHINESE POSTMAN is still a hard problem when each priority class is connected and their number is constant or small, that is, whether HIERARCHICAL CHINESE POSTMAN is in XP or FPT with respect to the parameter "number of priority classes." We conjecture that HIERARCHICAL CHINESE POSTMAN is indeed in XP. Note that, by the above reduction from RURAL POSTMAN, solving the FPT question in the affirmative would also imply that RURAL POSTMAN is FPT with respect to the number of connected components in the required edges. This is an intriguing and currently open question (see Challenge 4).

Tractability with linear precedence relations. If one is given a *linear* instead of a partial ordering of the edge partition, then HIERARCHICAL CHINESE POSTMAN with connected priority classes becomes polynomial-time solvable [27, 43, 62]. The central idea to get a polynomial-time algorithm here is to consider for each of the partition classes the subproblem of traversing every one of its edges. Dror, Stern, and Trudeau [27] showed that the optimal route that enters such a partition class at a given vertex u, traverses every one of its edges, and then leaves the class at another given vertex v can be computed in polynomial time. Call u the *entry point* and v the *exit point*. One now has to find a pair of a "good" entry point and a "good" exit point for each of the edge partition classes. Dror, Stern, and Trudeau [27] solved this problem by modeling it as a shortest-path problem on a graph whose vertices correspond to pairs of entry and exit points. Their algorithm gives a running time upper bound of $O(k|V|^5)$. More recently, alternative algorithms

with running times $O(k^3|V|^3)$ and $O(k|V|^4)$ have been given by Ghiani and Improta [43] and Korteweg and Volgenant [62], respectively.

Korteweg and Volgenant [62] also showed that the complexity upper bound of $O(k|V|^4)$ holds for two modifications of HIERARCHICAL CHINESE POSTMAN with a linear precedence relation and connected priority classes: One variant distinguishes between servicing and traversing costs such that only servicing has to respect the precedence relation, and the other variant seeks to minimize the completion cost of the first priority class, then the completion cost of the second priority class, and so on, instead of minimizing the overall tour cost.

2.2.5 ▪ Multiple postmen

A natural variant of CHINESE POSTMAN is to search for k tours in an edge-weighted graph G, such that each edge in G is traversed by at least one of the postmen. More formally, this can be stated as the following problem with objective function ω:

MIN-ω-k-CHINESE POSTMAN

Input: A connected, undirected graph $G = (V, E)$ with cost $c(e) \geq 0$ for every $e \in E$ and a maximum cost $c_{\max}$. Furthermore, a distinguished vertex $v \in V$ (the post office).

Question: Are there k tours T^* each starting and ending in v, such that none of the tours are empty, each edge in E is traversed by at least one of the tours, and $\omega(T^*) \leq c_{\max}$?

When focusing on the undirected case, the complexity of k-CHINESE POSTMAN highly depends on which objective function ω we choose: Minimizing the sum of the tour costs yields the polynomial-time solvable MIN-SUM-k-CHINESE POSTMAN [80, 102], whereas minimizing the maximum tour cost yields the NP-hard MIN-MAX-k-CHINESE POSTMAN [40]. This also holds if we substitute a directed graph for G. If both arcs and edges are present, then, clearly, both variants have MIXED CHINESE POSTMAN as special case and are NP-hard.

Minimizing the sum of the tour costs. Polynomial-time solvability of MIN-SUM-k-CHINESE POSTMAN has been proved by Zhang [102] and, independently, by Pearn [80], who observed that a lower bounding procedure for CAPACITATED ARC ROUTING proposed by Benavent et al. [10] solves MIN-SUM-k-CHINESE POSTMAN optimally. The main observation is that, if there is a postman tour for one postman that visits the post office v at least k times, then it can be split into k tours of the same cost. The converse is also true. Using this observation, one can solve MIN-SUM-k-CHINESE POSTMAN by introducing copies of the post office v as needed and then applying a matching technique similar to the one for the classical CHINESE POSTMAN problem. DIRECTED MIN-SUM-k-CHINESE POSTMAN can be solved using similar ideas [80, 102]. Pearn [80] also observed that some polynomial-time solvable special cases of MIXED CHINESE POSTMAN and WINDY CHINESE POSTMAN transfer to their k postmen variants.

It is interesting to note that the post office v is crucial to the polynomial-time solvability of undirected MIN-SUM-k-CHINESE POSTMAN. In fact, the problem becomes NP-hard if there is an unbounded number of postmen, and if there is no post office, that is, each tour can start and end at an arbitrary vertex of the graph. Thomassen [91] obtained this hardness result for the undirected case by reduction from a graph coloring problem. Simpler NP-hardness reductions for both directed and undirected input graphs were given more recently by Gutin, Muciaccia, and Yeo [52].

Of course, having an unbounded number of postmen is not realistic in many practical scenarios. Fortunately, MIN-SUM-k-CHINESE POSTMAN without a post office becomes tractable if the number of postmen is small: Gutin, Muciaccia, and Yeo [52] described how to compute a problem kernel with $O(k^2 \log k)$ vertices in the undirected case. Their procedure either obtains k tours of the input graph or, if this is not possible, guarantees that there are only a few vertices of degree at least three or degree one. Vertices of degree two can then be removed by a simple reduction rule that shortens long paths, preserving k tours if they are present. This then yields a bound on all vertices in the problem kernel. Using an exhaustive search procedure on the problem kernel then gives an FPT algorithm for finding the k tours. The directed case seems more complicated, but also this case is FPT with respect to k: Using a theorem from Ramsey theory, Gutin et al. [51] observed that either one can extract k tours or the input graph has some specific structure that lends itself to dynamic programming; in this way they obtained an FPT algorithm.[2] The dependence of the running time on the parameter k is large, however, and it remains to engineer algorithms with faster running times.

Minimizing the maximum tour cost. Frederickson, Hecht, and Kim [40] showed that MIN-MAX-k-CHINESE POSTMAN is NP-hard via a simple reduction from k-PARTITION. Intuitively, the reduction exploits that a difficulty in solving MIN-MAX-k-CHINESE POSTMAN lies in deciding which arc is to be traversed by which vehicle. An analogous argument also holds if the input graph is directed. On the positive side, Ahr [3] observed that MIN-MAX-k-CHINESE POSTMAN is polynomial-time solvable on paths, cycles, and cliques. It would be interesting to know whether his results can be generalized to broader graph classes. Motivated by the NP-hardness, Frederickson, Hecht, and Kim [40] gave a factor $(2-1/k)$ approximation algorithm that runs in $O(|V|^3)$ time. The basic idea is to first find a postman tour for a single postman and then split it into k tours in a clever way.

2.2.6 ▪ Further variants

We now briefly state some results for further interesting variants of CHINESE POSTMAN.

Turn constraints. In some practical applications, it is necessary to forbid certain ways to traverse crossings. For example, this is due to traffic rules or, in the case of snow plowing, to avoid depositing snow on the crossing. Unfortunately, such turn constraints make CHINESE POSTMAN NP-hard, as proved by Benavent and Soler [11].

Traversing at least one edge out of a subset. A slight generalization of the constraint that each edge in the graph has to be traversed makes CHINESE POSTMAN NP-hard: Given a number of subsets of the edges, traverse at least one edge of each subset. This was observed by Dror and Haouari [26].

Maximizing the benefit from servicing streets. It might be beneficial to service a street multiple times, for example, when routing street sweepers. This can be modeled as another variant of CHINESE POSTMAN, but as Pearn and Wang [82] proved, this also results in an NP-hard problem.

[2] In an earlier version of this chapter, the authors asked whether MIN-SUM-k-CHINESE POSTMAN without a post office is FPT with respect to k. This question was also featured in a survey talk by Sorge [86]. Subsequently, Gutin et al. [51] resolved the challenge, showing fixed-parameter tractability.

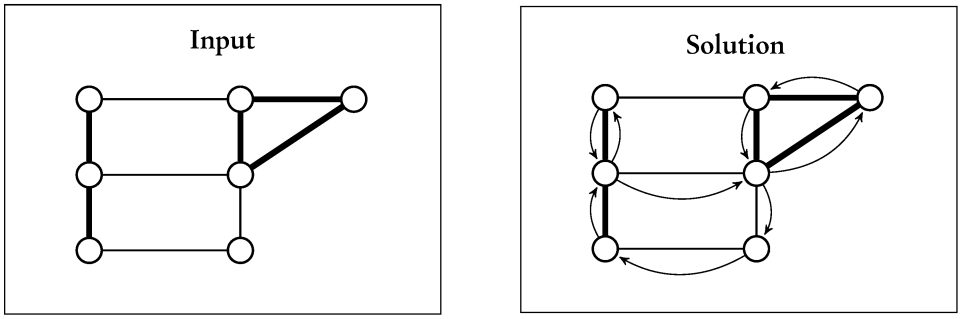


Figure 2.5. *Instance and solution for the* RURAL POSTMAN *problem.*

2.3 ▪ The Rural Postman Problem

The CHINESE POSTMAN problem we considered in the previous section can be used to model the task of delivering mail by some postman to houses in each street. In rural areas, however, not all streets may have houses to deliver mail to. If we take into account that a postman can use those streets to get his deliveries done more quickly, we end up with the following problem:

RURAL POSTMAN
Input: A connected (undirected) graph $G = (V,E)$ with edge costs $c(e) \geq 0$ for each $e \in E$, a set $R \subseteq E$ of *required edges*, and a maximum cost $c_{\max}$.
Question: Is there a tour in G that traverses every edge in R at least once and costs at most $c_{\max}$?

An input graph for RURAL POSTMAN with required edges drawn boldly and edge costs proportional to edge lengths in the drawing is depicted on the left in Figure 2.5, and a corresponding minimum-cost solution is pictured on the right.

Table 2.2 summarizes known results in approximation and parameterized complexity, omitting classical complexity, since all considered variants of RURAL POSTMAN are NP-complete if not *all* edges are required, which would make RURAL POSTMAN equivalent to CHINESE POSTMAN.

2.3.1 ▪ Classical complexity

By reducing the NP-hard HAMILTONIAN CYCLE problem to RURAL POSTMAN, Lenstra and Rinnooy Kan [64] showed RURAL POSTMAN to be NP-hard. HAMILTONIAN CYCLE asks whether a given graph admits a closed walk visiting each vertex exactly once. The reduction consists of attaching a loop to each vertex in the given graph $G = (V,E)$ and requiring a visit to all loops. All edges are given cost one, and the total cost of the desired tour is $2|V|$. If loops are not allowed in the graph, then we can simply subdivide them, that is, replace each loop by two edges and join them to a new vertex, and set the total cost of the desired tour to $3|V|$. It is not hard to see that removing the required edges from the postman tour leaves a tour of cost $|V|$ visiting all vertices of G at least once. Since all edges have cost one, no such tour visits a vertex twice. Similarly, HAMILTONIAN CYCLE on directed graphs can be reduced to RURAL POSTMAN on directed graphs, and, thus, NP-hardness for directed, mixed, and windy versions of RURAL POSTMAN follows from this proof.

Postman problems are very similar to graph problems related to the Eulerian property. In fact, a closed walk (such as a postman tour) in a graph can be seen as an Eulerian

Table 2.2. *Complexity of* RURAL POSTMAN (RP) *with respect to the parameters:* $|R|$ *is the number of required arcs,* γ *is the number of connected components induced by the required arcs,* b *is the number of imbalanced vertices,* $c_{\max}$ *is the maximum allowed tour cost, and* k *is the number of nonrequired arcs in a solution postman tour.*

RP variant	Approximation
Undirected	$O(\|V\|^3)$-time factor 3/2 [9, 39, 13] $O(\|V\|^3)$-time factor 3/2 for more general problem [57, 13]
Mixed	Require the triangle inequality to hold: factor 15/8 for symmetric [21] factor 9/5 if exactly the directed arcs are required [40]
RP variant	**Parameterized complexity**
Undirected	$O(\|V\|^{2\gamma}/\gamma! \cdot n)$-time algorithm [38] $2\|R\|$-vertex problem kernel $O(2^{3\|R\|} \cdot \|R\|^2 + \|V\|^3)$-time algorithm $O(2^\gamma \cdot (c_{\max} + \|V\|)^d)$-time randomized algorithm, $d = \text{const.}$ [53] results for directed graphs conjectured to hold [23, 87]
Directed	$O(4^k \cdot \|V\|^3)$-time algorithm [23] $O(4^{\gamma \log(b\gamma^2)} \cdot \|V\|^4)$-time algorithm [87] $O(2^\gamma \cdot (c_{\max} + \|V\|)^d)$-time randomized algorithm, $d = \text{const.}$ [53] no polynomial-size problem kernel with respect to γ and k [87]

multigraph. Thus, RURAL POSTMAN can be transformed into the following problem and vice versa by considering the required edges as edges of a multigraph that is to be extended to form an Eulerian cycle (see also Höhn, Jacobs, and Megow [55] and Dorn et al. [23]):

EULERIAN EXTENSION
Input: A multigraph $G = (V, E)$ with costs $c(\{u, v\}) \geq 0$ for all $u, v \in V$ and a maximum cost $c_{\max}$.
Question: Is there a multiset E' of pairs $\{u, v\}$ with $u, v \in V$ such that adding the edges in E' to G makes G Eulerian and $c(E') \leq c_{\max}$?

Höhn, Jacobs, and Megow [55] considered RURAL POSTMAN as EULERIAN EXTENSION on a special class of directed graphs where vertices are pairs of numbers and inserting an arc from (x_1, y_1) to (x_2, y_2) is only possible if $x_1 \leq x_2$ and $y_1 \leq y_2$. They called this variant TWO-DIMENSIONAL EULERIAN EXTENSION and proved it to be NP-hard.

A special case of RURAL POSTMAN on mixed graphs is the STACKER CRANE problem, which requires visiting all directed arcs of the input mixed graph. Frederickson, Hecht, and Kim [40] showed that also this problem remains NP-hard.

2.3.2 ▪ Approximability

Despite the fact that the closely related TRAVELING SALESMAN problem presumably cannot be approximated to within any constant factor of the optimum, constant-factor approximations for special cases of TRAVELING SALESMAN serve as the basis for constant-factor approximation algorithms for RURAL POSTMAN. For example, the famous algorithm by Christofides [20] approximates TRAVELING SALESMAN to within a factor

of 3/2 on graphs with triangle inequality (the input is required to be a complete graph such that, for each of three vertices u, v, and w, the inequality $c(\{u,v\})+c(\{v,w\}) \geq c(\{u,w\})$ holds). A similar approach is also used by Frederickson [39] and Jansen [57] to obtain constant-factor approximations for RURAL POSTMAN on instances respecting the triangle inequality. As pointed out by van Bevern et al. [13], this is sufficient for obtaining constant-factor approximations also for instances without triangle inequality.

In particular, polynomial-time algorithms of Frederickson [39] and Jansen [57] compute postman tours that are at most 50% longer than optimal tours. Both algorithms are based on the above-mentioned $O(|V|^3)$-time factor 3/2 approximation algorithm of Christofides [20]. While Frederickson's algorithm [39] is designed for RURAL POSTMAN, Jansen's algorithm [57] works also for the GENERAL ROUTING problem, where, aside from required edges that have to be visited at least once, we are also given a set of required vertices, each of which has to be visited *exactly* once.

In the following, we briefly sketch an adaptation of Jansen's algorithm [57] which is simplified and fitted to approximate RURAL POSTMAN to within a factor of 3/2. Given a complete graph $G = (V,E)$ with edge costs $c(\{u,v\}) \geq 0$ for each $u,v \in V$, and a set $R \subseteq E$ of required edges, it computes an Eulerian multigraph containing all edges of R. To this end, it connects the connected components induced by R in a treelike manner and then matches odd-degree vertices, such that inserting paths between them requires minimum additional cost. Next, we describe this four-step algorithm in more detail:

Step 1: Compute the graph $G_R := (\bigcup_{e \in R} e, R)$, and determine its connected components $\mathscr{C} := \{C_1, C_2, \ldots\}$.

Step 2: Compute the complete graph K on the vertex set $\mathscr{C}$ such that for each $i \neq j$ the cost of the edge $\{C_i, C_j\}$ equals the cost of a cheapest edge in G between any vertex of C_i and any vertex of C_j.

Step 3: Compute a minimum-cost spanning tree of K, and let G'_R denote the multigraph that results from adding to G_R the edges of G corresponding to the edges of the computed spanning tree.

Step 4: With V_{odd} denoting the vertices that have odd degree in G'_R, compute a minimum-cost matching M in $G[V_{\text{odd}}]$, and add the edges of M to G'_R.

Theorem 2.3 (Frederickson [39], Jansen [57]). RURAL POSTMAN *on undirected graphs can be approximated to within a factor of* 3/2 *in* $O(|V|^3)$ *time.*

In the following, we briefly sketch the proof of Theorem 2.3. For ease of presentation, we assume the input graph satisfies the triangle inequality, but, as pointed out by van Bevern et al. [13], the theorem also holds without triangle inequality.

Let c_{opt} denote the cost of an optimal solution for the given instance, and let $c(G'_R)$ and $c(M)$ denote the sum of costs of all edges in G'_R and in the matching M, respectively. Then, $c(G'_R) \leq c_{\text{opt}}$ since each optimal solution spans all connected components of G_R and visits all required edges, which cannot be done more cheaply than at cost $c(G'_R)$. Furthermore, one can show that the cost of the minimum-cost matching M in $G[V_{\text{odd}}]$ is at most $c_{\text{opt}}/2$: Since a vertex with odd degree in G'_R is also present in G_R, an optimal solution also visits all vertices in V_{odd}. Modifying an optimal tour by skipping over all vertices that are not in V_{odd}, we get a traveling salesperson tour in $G[V_{\text{odd}}]$, where *skipping over* v means replacing a subpath (u,v,w) of the tour by the subpath (u,w). Since the triangle inequality holds, the cost of this tour is at most c_{opt}. Furthermore, this tour

can be partitioned into two matchings; the smaller one costs at most $c_{\text{opt}}/2$. Since M is a minimum-cost matching, the claimed factor of 3/2 follows. The running time of the algorithm is dominated by the computation of the edge costs of K in Step 2. This can be done in $O(|V|^3)$ time using the Floyd–Warshall algorithm. It is unknown whether RURAL POSTMAN on directed graphs can be approximated to within a constant factor, even if the input respects the directed triangle inequality.

For solving RURAL POSTMAN on *symmetric* mixed graphs, where for each arc (u, v) in the input, there is also an arc (v, u) with equal cost, Chyu [21] developed an algorithm achieving an approximation factor of 15/8 if the triangle inequality holds. STACKER CRANE (the input is a mixed graph, and the required arcs are exactly the directed arcs of the input) can be approximated within a factor of 9/5 if the triangle inequality holds, as shown by Frederickson, Hecht, and Kim [40]. Although the latter is also based on Christophides's algorithm, it loses a factor of 6/5 compared to the factor 3/2 that is achieved by Frederickson's [39] and Jansen's [57] algorithms. It is interesting to investigate whether this deficiency can be overcome.

Challenge 3. *Can the gap between the approximation factors of Christophides's algorithm and known approximation algorithms for the mentioned special cases of* MIXED RURAL POSTMAN *be overcome? Is* MIXED RURAL POSTMAN, *in general, constant-factor approximable?*

2.3.3 ▪ Parameterized complexity

As exhibited in the previous section, where an approximation algorithm for TRAVELING SALESMAN was adapted to work for RURAL POSTMAN, node routing algorithms can help to solve arc routing problems: Pearn, Assad, and Golden [81] showed that the former can easily be transformed into the latter. Since this transformation of RURAL POSTMAN to TRAVELING SALESMAN creates instances with at most $3|R|$ vertices, it can be used for parameterized algorithms by combining it with the well-known dynamic programming algorithm by Held and Karp [54], yielding an algorithm that solves RURAL POSTMAN in $O(2^{3|R|} \cdot |R|^2 + |V|^3)$ time, where R is the set of required edges. RURAL POSTMAN also admits a simple problem kernel containing $2|R|$ vertices which can be constructed as follows: First, for all vertex pairs $\{u, v\}$, set $c'(\{u, v\})$ to the cost of a shortest path between u and v in the input graph. Then, for all $\{u, v\} \in R$, decrease $c_{\max}$ by $c(\{u, v\}) - c'(\{u, v\})$, remove all vertices that are not incident with required edges, and insert all edges $\{u, v\}$ for which $c'(\{u, v\}) \neq \infty$. Since only vertices that are incident with required edges remain, the constructed graph together with the cost function c' and the modified $c_{\max}$ is a problem kernel containing at most $2|R|$ vertices.

Normally, solution tours for RURAL POSTMAN traverse additional, nonrequired arcs to connect the required arcs. The number of additional arcs needed in an optimal solution can be considered dual to the number of required arcs. The number of additional arcs also fits the profile of an "above guarantee" parameter (such parameters are discussed by Mahajan and Raman [68] and Mahajan, Raman, and Sikdar [69]), since each solution is guaranteed to contain the arcs of R. Considering this parameterization, Dorn et al. [23] developed a dynamic programming algorithm that, given a directed instance of RURAL POSTMAN and an integer k, computes the minimum cost of a tour visiting all required arcs plus at most k additional arcs. The algorithm runs in $O(4^k \cdot |V|^3)$ time.

Theorem 2.4 (Dorn et al. [23]). RURAL POSTMAN *is FPT with respect to the number k of arcs that an optimal solution visits in addition to the required arcs.*

The algorithm of Dorn et al. is tailored to solve the EULERIAN EXTENSION problem for directed multigraphs with arc-insertion costs, which is equal to the RURAL POSTMAN problem. In the EULERIAN EXTENSION instance, the required arcs form connected components. The idea of the algorithm for EULERIAN EXTENSION is to make copies of these connected components such that an optimal solution can be assumed to visit each copy exactly once. Then, an optimal solution can be computed using dynamic programming, much like the algorithm of Held and Karp [54] for TRAVELING SALESMAN.

Pondering the NP-hardness proof of RURAL POSTMAN by Lenstra and Rinnooy Kan [64], one quickly notices that it produces very specific instances. Notably, the number γ of connected components induced by the required arcs is large, making the number of connected components a critical parameter for complexity analysis, as already implied by Orloff [77] in 1976. While Frederickson [38] gave an algorithm that runs in $O(|V|^{2\gamma}/\gamma! \cdot |V|)$ time, it is yet unknown whether RURAL POSTMAN is FPT with respect to γ.

Challenge 4. *Is* RURAL POSTMAN *FPT with respect to the number γ of connected components induced by the required arcs?*

Sorge et al. [87, 88] attacked this problem, showing that RURAL POSTMAN with respect to the parameter γ is "FPT-equivalent" to an unstudied, yet promising, NP-hard matching variant. As a side-result, Sorge et al. obtained an algorithm that solves RURAL POSTMAN in $O(4^{\gamma \log(b\gamma^2)} \cdot |V|^4)$ time, where b denotes the number of imbalanced vertices, that is, vertices whose indegree does not equal their outdegree. More recently, Gutin, Wahlström, and Yeo [53] and Marx and Pilipczuk [72] independently gave randomized algorithms for the above-mentioned matching variant. Both represent the instances as a potentially large polynomial in which certain coefficients indicate whether a solution exists or not. Testing for these coefficients can then be done in FPT time by using dynamic programming and exploiting the Schwartz–Zippel lemma.[3] The resulting algorithms may fail to recognize a solution, but the probability for this can be made arbitrarily small with reasonable running time overhead. Gutin, Wahlström, and Yeo [53] also applied the approach sketched above directly to RURAL POSTMAN; from their proof, one obtains the following.

Theorem 2.5 (Gutin, Wahlström, and Yeo [53]). RURAL POSTMAN *can be solved in $O(2^\gamma \cdot (c_{\max} + |V|)^{\ell+d})$ time for a given real $\ell > 0$ and some constant $d > 0$, reporting a yes-instance as a no-instance with probability at most $1/(c_{\max} + |V|)^\ell$.*

Their approach works for both directed and undirected variants of RURAL POSTMAN. As derandomizations for the above type of algorithm seem currently elusive, Challenge 4 remains an interesting open problem.

Regarding preprocessing of RURAL POSTMAN instances, Sorge et al. [87] showed that, unless the polynomial hierarchy collapses to the second level, there is no polynomial-size problem kernel with respect to the parameter k as defined above. Since $k \geq \gamma$, this result also holds for the parameter γ. However, considering the practical relevance of polynomial-time preprocessing, it is important to find a parameter for which, even in combination with γ, a problem kernel can be constructed.

[3] The Schwartz–Zippel lemma asserts that, when assigning random numbers to the variables in a nonzero polynomial, it is unlikely to evaluate to 0.

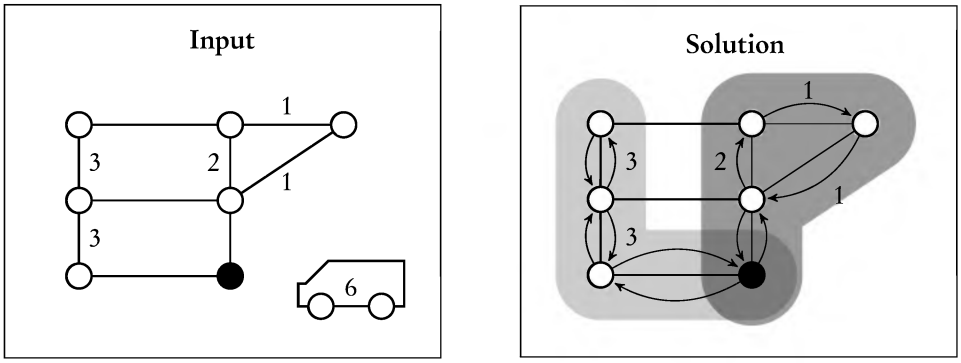


Figure 2.6. *Instance and solution for the* CAPACITATED ARC ROUTING *problem.*

2.4 ▪ The Capacitated Arc Routing Problem

The most general arc routing problem considered in this chapter is CAPACITATED ARC ROUTING, introduced by Golden and Wong [44]. It generalizes both of the previously considered problems, CHINESE POSTMAN and RURAL POSTMAN, and, speaking in terms of the mail delivery example, adds the possibility of serving mail to a set of required streets by multiple vehicles, each of which can only carry a limited number of parcels. In CAPACITATED ARC ROUTING, we are given an undirected graph, each edge of which is associated with a cost and some edges with a demand and a special vertex, called the *depot*. The depot houses an unlimited number of vehicles, each of a given capacity, that can *serve* edges or merely *traverse* them. An example is shown in the left part of Figure 2.6: The depot vertex v_0 is filled in black, edge demands are shown as labels next to the edges, and edge costs are proportional to the edge lengths in the drawing. All vehicles have capacity six. The task is then to serve each edge according to its demand by exactly one vehicle, where the sum of the demands of edges served by a single vehicle may not exceed its capacity. Moreover, the overall cost of the traversed edges is to be minimized. On the right in Figure 2.6 an optimal solution is shown. The route of one vehicle is shown on light gray background; the route of the other vehicle on dark gray background. Formally, the problem can be posed as follows:

CAPACITATED ARC ROUTING
Input: An undirected graph $G = (V, E)$ with a vehicle depot vertex $v_0 \in V$, with edge costs $c(e) \geq 0$ and edge demands $d(e) \geq 0$ for every $e \in E$, a vehicle capacity W, and a maximum cost $c_{\max}$.
Question: Is there a set $\mathcal{C}$ of cycles in G, each corresponding to the route of one vehicle and each passing through the depot vertex v_0, such that

1. $\sum_{C \in \mathcal{C}} \sum_{e \in C} c(e) \leq c_{\max}$,
2. each edge e with $d(e) > 0$ is traversed by at least one cycle and *served* by exactly one vehicle, and
3. the sum of demands of edges served by a single vehicle is at most W?

By giving each edge a demand of 1 and letting vehicles have a capacity equal to the number of edges in the graph, we obtain CHINESE POSTMAN. Giving only a subset of edges a demand of 1, we obtain RURAL POSTMAN.

2.4.1 ▪ Classical complexity

Since CAPACITATED ARC ROUTING can model RURAL POSTMAN, hardness results like the NP-hardness for the latter also apply to the former. However, using a different approach to showing the NP-hardness of CAPACITATED ARC ROUTING, one can observe that not only the underlying routing problem is hard. Independently from the structure of the underlying graph, CAPACITATED ARC ROUTING also contains the NP-hard problem component of distributing edge demands among vehicles. This can be seen by a simple polynomial-time many-one reduction from the NP- hard BIN PACKING problem:

BIN PACKING
Input: A bin size W, a number of bins B, and a list $w_1, \ldots, w_n$ of integers.
Question: Is there a partition $S_1 \cup \cdots \cup S_B$ of $\{w_1, \ldots, w_n\}$ such that $\sum_{w \in S_k} w \leq W$ for all $1 \leq k \leq B$?

To reduce BIN PACKING to CAPACITATED ARC ROUTING, an approach similar to the hardness proof of Golden and Wong [44] can be employed, where we here use a more restricted output graph structure to infer more hardness results: Simply create a path $(v_1, \ldots, v_{n+2})$, where v_{n+2} is the depot vertex. Let $d(\{v_{n+1}, v_{n+2}\}) = W$ and $c(\{v_{n+1}, v_{n+2}\}) = 1$. For all $k \leq n$, let $d(\{v_k, v_{k+1}\}) = w_k$ and $c(\{v_k, v_{k+1}\}) = 0$. It follows that the resulting graph can be served by $B+1$ vehicles of capacity W if and only if the given items $\{w_1, \ldots, w_n\}$ can be packed into B bins (one vehicle is needed to serve edge $\{v_{n+1}, v_{n+2}\}$). That is, the resulting graph can be served with cost $2(B+1)$ if and only if the input BIN PACKING instance is a yes-instance, since each vehicle has to traverse the edge $\{v_{n+1}, v_{n+2}\}$ twice. Combining this reduction with known hardness results for BIN PACKING by Jansen et al. [59], we obtain the following theorem.

Theorem 2.6. CAPACITATED ARC ROUTING *is* NP*-hard even if all edges are required to be visited, the requested maximum cost and number of required vehicles is a constant, and the maximum vertex degree in the graph is at most two.*

Theorem 2.6 contrasts the polynomial-time solvability of RURAL POSTMAN in the case that all edges are required. It is easy to execute the same reduction of BIN PACKING to CAPACITATED ARC ROUTING with directed graphs. Thus, the NP-hardness results hold in the same way for generalizations of CAPACITATED ARC ROUTING that allow for directed edges, multiple depots, or a requested time interval for each edge, or that allow forbidding certain turns of vehicles (for example, to properly remove snow from crossroads).

Relations to node routing problems. One approach to obtaining exact (rather than heuristic) solutions for CAPACITATED ARC ROUTING is to transform CAPACITATED ARC ROUTING into the CAPACITATED VEHICLE ROUTING problem, as defined by Dantzig and Ramser [22], and then apply state-of-the-art algorithms to the latter.

CAPACITATED VEHICLE ROUTING
Input: A set V of vertices with a vehicle depot vertex $v_0 \in V$, with costs $c(\{u,v\}) \geq 0$ for all $u,v \in V$ and vertex demands $d(v) \geq 0$ for every $v \in V$, a vehicle capacity W, and a maximum cost $c_{\max}$.
Question: Is there a partition $C_1 \uplus \cdots \uplus C_\ell = \{v \in V \setminus \{v_0\} \mid d(v) > 0\}$ and for each C_i a permutation σ_i such that, for $1 \leq i \leq \ell$, it holds that $\sum_{v \in C_\ell} d(v) \leq W$ and such that

$$\sum_{i=1}^{\ell} \left(c(\{v_0, \sigma_{i,1}\}) + c(\{v_0, \sigma_{i,|C_i|}\}) + \sum_{j=1}^{|C_i|-1} c(\{\sigma_{i,j}, \sigma_{i+1,j}\}) \right) \leq c_{\max}?$$

In order to solve CAPACITATED ARC ROUTING using algorithms for CAPACITATED VEHICLE ROUTING, the first transformation from CAPACITATED ARC ROUTING to CAPACITATED VEHICLE ROUTING by Pearn, Assad, and Golden [81] has been subsequently improved to yield smaller instances by Baldacci and Maniezzo [7] and by Longo, de Aragão, and Uchoa [67].

Golden and Wong [44] give a straightforward transformation from CAPACITATED VEHICLE ROUTING to CAPACITATED ARC ROUTING, which, however, requires the input VEHICLE ROUTING instance to respect the triangle inequality: Given a VEHICLE ROUTING instance, split each vertex with positive demand into two, and join them by a zero-cost edge whose demand corresponds to the demand of the original vertex. If the input VEHICLE ROUTING instance respects the triangle inequality, this transformation is correct since a solution for the output CAPACITATED ARC ROUTING instance can be transformed into an equal-cost solution of the original VEHICLE ROUTING instance. Since a vertex in VEHICLE ROUTING may be visited at most once, whereas the found CAPACITATED ARC ROUTING solution may visit the corresponding edges several times, this transformation is not correct *in general*. However, if the triangle inequality of the VEHICLE ROUTING instance holds, then tours that visit a vertex twice can be shortened.

2.4.2 ▪ Approximability

The NP-hardness of CAPACITATED ARC ROUTING even in very special cases tempts us to search for approximate solutions. However, using a reduction from PARTITION similar to the proof of Theorem 2.6, Golden and Wong [44] showed that computing a factor $3/2$ approximation for CAPACITATED ARC ROUTING is NP-hard, even if the cost function respects the triangle inequality. On the positive side, there are factor $(7/2 - 3/W)$ approximations for CAPACITATED ARC ROUTING by Jansen [58] and Wøhlk [94].

Theorem 2.7 (Golden and Wong [44], Jansen [58], Wøhlk [94], van Bevern et al. [13]). *Computing a factor* $3/2$ *approximation for* CAPACITATED ARC ROUTING *is* NP*-hard. A factor* $(7/2 - 3/W)$ *approximation can be computed in polynomial time.*

In fact, the algorithms by Jansen [58] and Wøhlk [94] assume the triangle inequality to hold. Moreover, in the literature, the following claim can also be found [44, 93, 94]: If the cost function in a CAPACITATED ARC ROUTING instance is not required to respect the triangle inequality, then computing a factor α approximation for CAPACITATED ARC ROUTING is NP-hard for any $\alpha > 0$. However, van Bevern et al. [13] showed that any factor α approximation for CAPACITATED ARC ROUTING with triangle inequality also yields a factor α approximation for CAPACITATED ARC ROUTING without triangle

inequality. Thus, the approximation factor given in Theorem 2.7 can also be achieved without the triangle inequality.

It is unknown whether CAPACITATED ARC ROUTING on directed graphs is constant-factor approximable. Herein, it would be sufficient to find a polynomial-time constant-factor approximation for the case where the directed triangle inequality holds [13].

Challenge 5. *Find a constant-factor approximation for* CAPACITATED ARC ROUTING *on directed graphs, or prove that any constant-factor approximation would imply* P = NP.[4]

2.4.3 ▪ Parameterized complexity

The parameterized complexity of CAPACITATED ARC ROUTING is still an untouched field. If we do not require that the input respect the triangle inequality, then Theorem 2.6 rules out the parameters "maximum cost $c_{\max}$," pathwidth, maximum number of vehicles, and maximum degree of the input graph with regard to fixed-parameter tractability, since CAPACITATED ARC ROUTING remains NP-hard even if all these parameters are constant. From parameterized considerations about BIN PACKING by Jansen et al. [59] and Theorem 2.6, we obtain that, even if we were to consider the demands of edges to be encoded in unary, CAPACITATED ARC ROUTING is $W[1]$-hard with respect to the parameter $c_{\max}$ or, analogously, with respect to the number of vehicles. It is yet unclear whether, with respect to these parameters, CAPACITATED ARC ROUTING is in XP.

Challenge 6. BIN PACKING *is in* XP *but W[1]-hard with respect to the parameter "number of required bins" if the input integers are encoded in unary [59]. This implies W[1]-hardness for* CAPACITATED ARC ROUTING *with respect to the maximum cost $c_{\max}$ if the edge demands are encoded in unary. Is* CAPACITATED ARC ROUTING *in* XP *under these conditions?*

If we require that the input satisfy the triangle inequality, then the NP-hardness proof initially given by Golden and Wong [44] shows that CAPACITATED ARC ROUTING even remains NP-hard on graphs of diameter one (the diameter of a graph is the maximum length of a shortest path between two vertices). Hence, the single parameter "diameter" is also ineffective. In their reduction, the parameters "maximum cost $c_{\max}$" and "number of required vehicles" also remain constant. In contrast, the parameters "maximum degree" and "treewidth" in the instances created by this reduction may be large, but, with respect to these parameters, we can find a trivial problem kernel: If the input CAPACITATED ARC ROUTING instance satisfies the triangle inequality, then it is a no-instance or every vertex in the input graph is adjacent to the depot vertex. This is because every vertex v has to be reachable from the depot vertex by some path, and, therefore, v has an edge to the depot vertex that has at most the length of this path. Since all vertices are adjacent to the depot vertex, they are also adjacent to each other, forming a clique. The number of vertices in the clique, and, therefore, the number of vertices in the input graph, is then bounded linearly by the maximum degree and the treewidth, yielding a problem kernel with a linear number of vertices.

It turns out that, with respect to most parameters, CAPACITATED ARC ROUTING remains hard or, if we require the triangle inequality, becomes trivial. Hence, one of the main challenges considering the parameterized complexity of CAPACITATED ARC

[4] In an earlier version of this chapter, the authors asked whether CAPACITATED ARC ROUTING is constant-factor approximable without triangle inequality. Subsequently, van Bevern et al. [13] answered this question affirmatively.

ROUTING seems to be the task of identifying meaningful and small parameters that make the problem tractable. Wøhlk [93] raises the point that restrictions on the demands may help in coping with the hardness of the problem: For example, on paths or cycles with uniform demands the problem can be solved in polynomial time. Also, the BIN PACKING problem used in the NP-hardness reduction for CAPACITATED ARC ROUTING above becomes polynomial-time solvable if the number of bins is a constant and the integers are at most polynomial in the input size.

2.5 ▪ Conclusion and outlook

Classifying the computational complexity of a problem lies at the heart of developing and also justifying lines of attack for solving it. Indeed, many methods for solving arc routing problems are based on heuristics and mathematical programming; this is justified by the fact that most of these problems are NP-hard. A proof of NP-hardness, however, does not mean the end of useful theoretical (and practically useful) work on the considered problem. That is, it may be possible to identify meaningful polynomial-time solvable special cases, to derive efficient approximation algorithms, or to identify problem-specific parameters to be exploited in the spirit of fixed-parameter tractability. Thus, throughout this chapter we featured several challenging open problems whose answering would help to obtain a further refined view on the computational complexity of important arc routing problems.

Clearly, the selected open problems could be extended by numerous ways of systematically identifying research challenges. Let us mainly focus on parameterized complexity analysis here, the "freshest approach" discussed in this chapter. A fruitful way of identifying parameters that have significant influence on the computational complexity of a problem is to revisit known hardness reductions proving NP-hardness or the like. In these proofs, one then checks whether they rely on certain parameter values that are necessarily unbounded to make the proof work. For instance, the NP-hardness proof for RURAL POSTMAN by Lenstra and Rinnooy Kan [64] heavily relies on the fact that the number of connected components is large. Indeed, Frederickson [38] showed the problem to be polynomial-time solvable for a constant number of connected components, rendering this a very interesting parameter and leading to the challenging open question whether RURAL POSTMAN is fixed-parameter tractable with respect to this parameter. A partial answer has been given by Gutin, Wahlström, and Yeo [53], who showed that a *randomized* FPT algorithm exists. Obviously, the same approach for identifying parameters can be pursued for other problems as well and also has been termed "deconstructing intractability" [32, 61, 76].

A further way to spot interesting parameterizations for arc routing problems is to inspect real-world data and to measure various network parameters (such as vertex degree, diameter, number of connected components, etc.) in order to see whether some of them adopt small values in applications. If so, then this strongly motivates parameterized complexity investigations with respect to the parameters identified in this way. This is also known as data-driven parameterization.

We believe that, although classical complexity classification of most arc routing problems is in a sense settled, the advent of parameterized complexity analysis paves the way for a prospective theory-based exploration of the computational complexity landscape of arc routing problems. With the research challenges featured in this chapter, we hope to help kick off such a development.[5]

[5]Indeed, after a survey talk by Sorge [86] based on an earlier version of this chapter, several new results on the parameterized complexity of arc routing problems appeared and were included [50, 51, 52, 53, 72].

Acknowledgments

We are grateful to an anonymous referee whose comments greatly helped to improve the presentation of this chapter, and we thank Greg N. Frederickson and Sanne Wøhlk for making their Ph.D. works accessible.

René van Bevern and Manuel Sorge gratefully acknowledge support provided by the DFG grant DAPA (NI 369/12), and Mathias Weller gratefully acknowledges support provided by the DFG grant DARE (NI 369/11). This work was performed while all authors were with Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany.

Bibliography

[1] *Clay Mathematics Institute*, 2013. http://www.claymath.org/millennium-problems.

[2] S. AARONSON, *The complexity zoo*, 2013. https://complexityzoo.uwaterloo.ca/Complexity_Zoo.

[3] D. AHR, *Contributions to Multiple Postmen Problems*, Ph.D. thesis, Ruprecht-Karls-Universität Heidelberg, Heidelberg, Germany, 2004.

[4] A.S. ALFA AND D.Q. LIU, *Postman routing problem in a hierarchical network*, Engineering Optimization, 14 (1988), pp. 127–138.

[5] S. ARORA AND B. BARAK, *Computational Complexity: A Modern Approach*, Cambridge University Press, Cambridge, UK, 2009.

[6] G. AUSIELLO, P. CRESCENZI, G. GAMBOSI, V. KANN, A. MARCHETTI-SPACCAMELA, AND M. PROTASI, *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*, Springer, New York, 1999.

[7] R. BALDACCI AND V. MANIEZZO, *Exact methods based on node-routing formulations for undirected arc-routing problems*, Networks, 47 (2006), pp. 52–60.

[8] E.J. BELTRAMI AND L.D. BODIN, *Networks and vehicle routing for municipal waste collection*, Networks, 4 (1974), pp. 65–94.

[9] E. BENAVENT, V. CAMPOS, A. CORBERÁN, AND E. MOTA, *Análisis de heurísticos para el problema del cartero rural*, Trabajos de Estadística y de Investigación Operativa, 36 (1985), pp. 27–38.

[10] ——, *The capacitated arc routing problem: Lower bounds*, Networks, 22 (1992), pp. 669–690.

[11] E. BENAVENT AND D. SOLER, *The directed rural postman problem with turn penalties*, Transportation Science, 33 (1999), pp. 408–418.

[12] C. BERGE, *Graphs*, North-Holland, Amsterdam, 1985.

[13] R. VAN BEVERN, S. HARTUNG, A. NICHTERLEIN, AND M. SORGE, *Constant-factor approximations for capacitated arc routing without triangle inequality*, Operations Research Letters, 42 (2014), pp. 290–292.

[14] H.L. BODLAENDER, *Kernelization: New upper and lower bound techniques*, in Proceedings of the 4th International Workshop on Parameterized and Exact Computation (IWPEC '09), Lecture Notes in Computer Science 5917, Springer, New York, 2009, pp. 17–37.

[15] H.L. BODLAENDER, R.G. DOWNEY, M.R. FELLOWS, AND D. HERMELIN, *On problems without polynomial kernels*, Journal of Computer and System Sciences, 75 (2009), pp. 423–434.

[16] H.L. BODLAENDER AND A.M.C.A. KOSTER, *Combinatorial optimization on graphs of bounded treewidth*, The Computer Journal, 51 (2008), pp. 255–269.

[17] L. CAI, *Parameterized complexity of vertex colouring*, Discrete Applied Mathematics, 127 (2003), pp. 415–429.

[18] J. CHEN, I.A. KANJ, AND G. XIA, *Improved upper bounds for vertex cover*, Theoretical Computer Science, 411 (2010), pp. 3736–3756.

[19] N. CHRISTOFIDES, *The optimum traversal of a graph*, Omega, 1 (1973), pp. 719–732.

[20] N. CHRISTOFIDES, *Worst case analysis of a new heuristic for the traveling salesman problem*, Management Science Research Rept. 388, Carnegie-Mellon University, Pittsburgh, PA, 1976.

[21] C.-C. CHYU, *A mixed-strategy heuristic for the mixed arc routing problem*, Journal of the Chinese Institute of Industrial Engineers, 18 (2001), pp. 68–76.

[22] G.B. DANTZIG AND J.H. RAMSER, *The truck dispatching problem*, Management Science, 6 (1959), pp. 80–91.

[23] F. DORN, H. MOSER, R. NIEDERMEIER, AND M. WELLER, *Efficient algorithms for Eulerian extension and rural postman*, SIAM Journal on Discrete Mathematics, 27 (2013), pp. 75–94.

[24] R.G. DOWNEY AND M.R. FELLOWS, *Fundamentals of Parameterized Complexity*, Springer, New York, 2013.

[25] M. DROR, *Arc Routing: Theory, Solutions, and Applications*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.

[26] M. DROR AND M. HAOUARI, *Generalized Steiner problems and other variants*, Journal of Combinatorial Optimization, 4 (2000), pp. 415–436.

[27] M. DROR, H. STERN, AND P. TRUDEAU, *Postman tour on a graph with precedence relation on arcs*, Networks, 17 (1987), pp. 283–294.

[28] J. EDMONDS, *The Chinese postman problem*, Operations Research, Supplement 1 (1975), pp. B73–B77.

[29] J. EDMONDS AND E.L. JOHNSON, *Matching, Euler tours and the Chinese postman*, Mathematical Programming, 5 (1973), pp. 88–124.

[30] H.A. EISELT, M. GENDREAU, AND G. LAPORTE, *Arc routing problems, part II: The rural postman problem*, Operations Research, 43 (1995), pp. 399–414.

[31] H.A. EISELT, M. GENDREAU, AND G. LAPORTE, *Arc routing problems, part I: The Chinese postman problem*, Operations Research, 43 (1995), pp. 231–242.

[32] M.R. FELLOWS, B.M.P. JANSEN, AND F.A. ROSAMOND, *Towards fully multivariate algorithmics: Parameter ecology and the deconstruction of computational complexity*, European Journal of Combinatorics, 34 (2013), pp. 541–566.

[33] C.G. FERNANDES, O. LEE, AND Y. WAKABAYASHI, *Minimum cycle cover and Chinese postman problems on mixed graphs with bounded tree-width*, Discrete Applied Mathematics, 157 (2009), pp. 272–279.

[34] J. FLUM AND M. GROHE, *Parameterized Complexity Theory*, Springer, New York, 2006.

[35] L.R. FORD AND D.R. FULKERSON, *Flows in Networks*, Princeton University Press, Princeton, NJ, 2010.

[36] L. FORTNOW, *The Golden Ticket: P, NP, and the Search for the Impossible*, Princeton University Press, Princeton, NJ, 2013.

[37] L. FORTNOW AND R. SANTHANAM, *Infeasibility of instance compression and succinct PCPs for NP*, Journal of Computer and System Sciences, 77 (2011), pp. 91–106.

[38] G.N. FREDERICKSON, *Approximation Algorithms for NP-hard Routing Problems*, Ph.D. thesis, Faculty of the Graduate School of the University of Maryland, College Park, MD, 1977.

[39] ——, *Approximation algorithms for some postman problems*, Journal of the ACM, 26 (1979), pp. 538–554.

[40] G.N. FREDERICKSON, M.S. HECHT, AND C.E. KIM, *Approximation algorithms for some routing problems*, SIAM Journal on Computing, 7 (1978), pp. 178–193.

[41] M.R. GAREY AND D.S. JOHNSON, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, San Francisco, CA, 1979.

[42] W.I. GASARCH, *Guest column: The second P =?NP poll*, ACM SIGACT News, 43 (2012), pp. 53–77.

[43] G. GHIANI AND G. IMPROTA, *An algorithm for the hierarchical Chinese postman problem*, Operations Research Letters, 26 (2000), pp. 27–32.

[44] B.L. GOLDEN AND R.T. WONG, *Capacitated arc routing problems*, Networks, 11 (1981), pp. 305–315.

[45] O. GOLDREICH, *P, NP, and NP-Completeness*, Cambridge University Press, Cambridge, UK, 2010.

[46] M. GUAN, *Graphic programming using odd or even points*, Chinese Mathematics, 1 (1962), pp. 273–277.

[47] ——, *On the windy postman problem*, Discrete Applied Mathematics, 9 (1984), pp. 41–46.

[48] J. GUO, F. HÜFFNER, AND R. NIEDERMEIER, *A structural view on parameterizing problems: Distance from triviality*, in Proceedings of the 1st International Workshop on Parameterized and Exact Computation (IWPEC '04), Lecture Notes in Computer Science 3162, Springer, New York, 2004, pp. 162–173.

[49] J. GUO AND R. NIEDERMEIER, *Invitation to data reduction and problem kernelization*, ACM SIGACT News, 38 (2007), pp. 31–45.

[50] G. GUTIN, M. JONES, AND B. SHENG, *Parameterized complexity of the k-arc Chinese postman problem*, Computing Research Repository, arXiv:1403.1512 (2014).

[51] G. GUTIN, M. JONES, B. SHENG, AND M. WAHLSTRÖM, *Parameterized directed k-Chinese postman problem and k arc-disjoint cycles problem on Euler digraphs*, in Proceedings of the 40th International Workshop on Graph-Theoretic Concepts in Computer Science (WG '14), Lecture Notes in Computer Science, Springer, 2014. Accepted for publication.

[52] G. GUTIN, G. MUCIACCIA, AND A. YEO, *Parameterized complexity of k-Chinese postman problem*, Theoretical Computer Science, 513 (2013), pp. 124–128.

[53] G. GUTIN, M. WAHLSTRÖM, AND A. YEO, *Parameterized rural postman and conjoining bipartite matching problems*, Computing Research Repository, arXiv:1308.2599 (2013).

[54] M. HELD AND R. M. KARP, *A dynamic programming approach to sequencing problems*, Journal of the Society for Industrial and Applied Mathematics, 10 (1962), pp. 196–210.

[55] W. HÖHN, T. JACOBS, AND N. MEGOW, *On Eulerian extensions and their application to no-wait flowshop scheduling*, Journal of Scheduling, 15 (2012), pp. 295–309.

[56] F. HÜFFNER, R. NIEDERMEIER, AND S. WERNICKE, *Techniques for practical fixed-parameter algorithms*, The Computer Journal, 51 (2008), pp. 7–25.

[57] K. JANSEN, *An approximation algorithm for the general routing problem*, Information Processing Letters, 41 (1992), pp. 333–339.

[58] K. JANSEN, *Bounds for the general capacitated routing problem*, Networks, 23 (1993), pp. 165–173.

[59] K. JANSEN, S. KRATSCH, D. MARX, AND I. SCHLOTTER, *Bin packing with fixed number of bins revisited*, Journal of Computer and System Sciences, 79 (2013), pp. 39–49.

[60] C. KOMUSIEWICZ AND R. NIEDERMEIER, *New races in parameterized algorithmics*, in Proceedings of the 37th International Symposium on Mathematical Foundations of Computer Science (MFCS '12), Lecture Notes in Computer Science 7464, Springer, New York, 2012, pp. 19–30.

[61] C. KOMUSIEWICZ, R. NIEDERMEIER, AND J. UHLMANN, *Deconstructing intractability—a multivariate complexity analysis of interval constrained coloring*, Journal of Discrete Algorithms, 9 (2011), p. 137–151.

[62] P. KORTEWEG AND T. VOLGENANT, *On the hierarchical Chinese postman problem with linear ordered classes*, European Journal of Operational Research, 169 (2006), pp. 41–52.

[63] J. VAN LEEUWEN AND R.B. TAN, *Compact routing methods: A survey*, in Proceedings of the 1st International Colloquium on Structural Information and Communication Complexity (SIROCCO '94), Carleton University Press, Ottawa, Ontario, Canada, 1994, pp. 99–110.

[64] J.K. LENSTRA AND A.H.G. RINNOOY KAN, *On general routing problems*, Networks, 6 (1976), pp. 273–280.

[65] Y. LIN AND Y. ZHAO, *A new algorithm for the directed Chinese postman problem*, Computers & Operations Research, 15 (1988), pp. 577–584.

[66] D. LOKSHTANOV, N. MISRA, AND S. SAURABH, *Kernelization—preprocessing with a guarantee*, in The Multivariate Algorithmic Revolution and Beyond—Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday, Lecture Notes in Computer Science 7370, Springer, New York, 2012, pp. 129–161.

[67] H. LONGO, M.P. DE ARAGÃO, AND E. UCHOA, *Solving capacitated arc routing problems using a transformation to the CVRP*, Computers & Operations Research, 33 (2006), pp. 1823–1837.

[68] M. MAHAJAN AND V. RAMAN, *Parameterizing above guaranteed values: MaxSat and MaxCut*, Journal of Algorithms, 31 (1999), pp. 335–354.

[69] M. MAHAJAN, V. RAMAN, AND S. SIKDAR, *Parameterizing above or below guaranteed values*, Journal of Computer and System Sciences, 75 (2009), pp. 137–153.

[70] J. MARKOFF, *Prizes aside, the P-NP puzzler has consequences*, The New York Times, October 7th, 2009. Available online at http://www.nytimes.com/2009/10/08/science/Wpolynom.html.

[71] D. MARX, B. O'SULLIVAN, AND I. RAZGON, *Finding small separators in linear time via treewidth reduction*, ACM Transactions on Algorithms, 9 (2013), p. 30.

[72] D. MARX AND M. PILIPCZUK, *Everything you always wanted to know about the parameterized complexity of subgraph isomorphism (but were afraid to ask)*, Computing Research Repository, arXiv:1307.2187 (2013).

[73] S. MICALI AND V. V. VAZIRANI, *An $O(\sqrt{|v|}|E|)$ algorithm for finding maximum matching in general graphs*, in Proceedings of the 27th Annual IEEE Symposium on Foundations of Computer Science (FOCS '86), IEEE, Washington, DC, 1980, pp. 17–27.

[74] E. MINIEKA, *The Chinese postman problem for mixed networks*, Management Science, 25 (1979), pp. 643–648.

[75] R. NIEDERMEIER, *Invitation to Fixed-Parameter Algorithms*, Oxford University Press, Oxford, UK, 2006.

[76] ——, *Reflections on multivariate algorithmics and problem parameterization*, in Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science (STACS '10), vol. 5 of LIPIcs, Schloss Dagstuhl—Leibniz-Zentrum fuer Informatik, 2010, pp. 17–32.

[77] C.S. ORLOFF, *On general routing problems: Comments*, Networks, 6 (1976), pp. 281–284.

[78] C.H. PAPADIMITRIOU, *On the complexity of edge traversing*, Journal of the ACM, 23 (1976), pp. 544–554.

[79] ——, *Computational Complexity*, Addison–Wesley, Reading, MA, 1994.

[80] W.L. PEARN, *Solvable cases of the k-person Chinese postman problem*, Operations Research Letters, 16 (1994), pp. 241–244.

[81] W.L. PEARN, A. ASSAD, AND B.L. GOLDEN, *Transforming arc routing into node routing problems*, Computers & Operations Research, 14 (1987), pp. 285–288.

[82] W.L. PEARN AND K.-H. WANG, *On the maximum benefit Chinese postman problem*, Omega, 31 (2003), pp. 269–273.

[83] N. PERRIER, A. LANGEVIN, AND J.F. CAMPBELL, *A survey of models and algorithms for winter road maintenance. Part IV: Vehicle routing and fleet sizing for plowing and snow disposal*, Computers & Operations Research, 34 (2007), pp. 258–294.

[84] B. RAGHAVACHARI AND J. VEERASAMY, *Approximation algorithms for the asymmetric postman problem*, in Proceedings of the 10th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA '99), SIAM, Philadelphia, 1999, pp. 734–741.

[85] B. RAGHAVACHARI AND J. VEERASAMY, *A 3/2-approximation algorithm for the mixed postman problem*, SIAM Journal on Discrete Mathematics, 12 (1999), pp. 425–433.

[86] M. SORGE, *Some algorithmic challenges in arc routing*, Talk at NII Shonan Seminar 18, Shonan, Japan, May 2013.

[87] M. SORGE, R. VAN BEVERN, R. NIEDERMEIER, AND M. WELLER, *From few components to an Eulerian graph by adding arcs*, in Proceedings of the 37th International Workshop on Graph-Theoretic Concepts in Computer Science (WG '11), Lecture Notes in Computer Science 6986, Springer, New York, 2011, pp. 307–319.

[88] ——, *A new view on rural postman based on Eulerian extension and matching*, Journal of Discrete Algorithms, 16 (2012), pp. 12–33.

[89] D.A. SPIELMAN AND S.-H. TENG, *Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time*, Journal of the ACM, 51 (2004), pp. 385–463.

[90] ——, *Smoothed analysis: An attempt to explain the behavior of algorithms in practice*, Communications of the ACM, 52 (2009), pp. 76–84.

[91] C. THOMASSEN, *On the complexity of finding a minimum cycle cover of a graph*, SIAM Journal on Computing, 26 (1997), pp. 675–677.

[92] V.V. VAZIRANI, *Approximation Algorithms*, Springer, New York, 2001.

[93] S. WØHLK, *Capacitated Arc Routing Problem, Theory and Solution*, Ph.D. thesis, University of Southern Denmark, Odense, Denmark, 2002.

[94] ——, *An approximation algorithm for the capacitated arc routing problem*, The Open Operational Research Journal, 2 (2008), pp. 8–12.

[95] D.P. WILLIAMSON AND D.B. SHMOYS, *The Design of Approximation Algorithms*, Cambridge University Press, Cambridge, UK, 2011.

[96] Z. WIN, *Contributions to Routing Problems*, Ph.D. thesis, Universität Augsburg, Augsburg, Germany, 1987.

[97] ——, *On the windy postman problem on Eulerian graphs*, Mathematical Programming, 44 (1989), pp. 97–112.

[98] F. J. ZARAGOZA MARTÍNEZ, *Postman Problems on Mixed Graphs*, Ph.D. thesis, University of Waterloo, Waterloo, Ontario, Canada, 2003.

[99] ——, *Complexity of the mixed postman problem with restrictions on the arcs*, in Proceedings of the 3rd International Conference on Electrical and Electronics Engineering (ICEEE'06), IEEE, Washington, DC, 2006, pp. 1–4.

[100] ——, *Complexity of the mixed postman problem with restrictions on the edges*, in Proceedings of the Seventh Mexican International Conference on Computer Science (ENC'06), IEEE, Washington, DC, 2006, pp. 3–10.

[101] ——, *Series-parallel graphs are windy postman perfect*, Discrete Mathematics, 308 (2008), pp. 1366–1374.

[102] L. ZHANG, *Polynomial algorithms for the k-Chinese postman problem*, in Proceedings of the IFIP 12th World Computer Congress on Algorithms, Software, Architecture—Information Processing '92, Volume 1, IFIP Transactions A-12, North-Holland, Amsterdam, 1992, pp. 430–435.

Chapter 3

The Undirected Chinese Postman Problem

Gilbert Laporte

3.1 ▪ Introduction

The *Chinese Postman Problem* (CPP) is arguably the most central problem in arc routing. In this chapter we review the undirected version of the CPP, as well as four of its variants: the generalized CPP, the cumulative CPP, the hierarchical CPP, and the CPP with time windows. Another variant, the CPP with profits, will be treated in Chapter 12.

3.2 ▪ The undirected Chinese Postman Problem

The undirected CPP takes its name from the Chinese mathematician Meigu Guan who introduced it more than 50 years ago (Guan [21]). The problem is defined on an undirected graph $G = (V, E)$, where $V = \{1, \ldots, n\}$ is the vertex set and $E = \{(i, j) : i, j \in V, i < j\}$ is the edge set. With each edge (i, j) is associated a travel cost c_{ij}. The problem is to determine a least-cost closed traversal of all edges of the graph. When G is connected and all vertices have an even degree, the graph is Eulerian (or unicursal), which means that there exists a CPP solution traversing each edge only once. As we saw in Chapter 1, Euler [14] observed that connectedness and evenness are two necessary conditions for unicursality, and Hierholzer [23] showed that these conditions were also sufficient.

In order for a feasible CPP solution to exist, the graph must of course be connected. The difficulty consists of determining an optimal solution when not all vertices have an even degree. The problem amounts to determining a least-cost *augmentation* of the graph that will render all degrees even, i.e., identifying a subset of edges that will be traversed more than once. In fact, it is never optimal to traverse the same edge more than twice since, if this were the case, pairs of traversals could be removed without disconnecting the graph and without changing degree parities. A least-cost augmentation is obtained by determining a minimum cost matching of the odd-degree vertices of G, where the matching cost of i and j is the cost of a shortest chain $(i, \ldots, j)$ in G.

The algorithm proposed by Guan [21] in his seminal paper effectively solves a matching problem while ensuring that the added chains do not intersect. Guan proved the following two necessary and sufficient conditions for optimality: (1) The solution has no

redundancy, which means that at most one edge linking two given vertices in the original graph is duplicated; (2) the cost of the added edges does not exceed half the solution cost. Note that this property lies at the heart of the Christofides [8] heuristic for the *Traveling Salesman Problem* (TSP), which has a worst-case performance ratio of 3/2 whenever the TSP cost matrix is symmetric and satisfies the triangle inequality.

A natural way of formulating the augmentation problem as a linear integer program is to define binary x_{ij} variables equal to 1 if and only if a copy of edge (i,j) appears in the augmentation. Let $T \subseteq V$ be the set of odd-degree vertices of V, and let $\delta(i)$ be the set of edges incident to vertex i. The formulation is then

$$\text{(CPP1)} \qquad \text{minimize} \sum_{(i,j)\in E} c_{ij}x_{ij} \tag{3.1}$$

$$\text{s.t.} \qquad \sum_{(i,j)\in\delta(i)} x_{ij} = \begin{cases} 1 \pmod 2 & \text{if } i \in T, \\ 0 \pmod 2 & \text{if } i \in V\backslash T, \end{cases} \tag{3.2}$$

$$x_{ij} = 0 \text{ or } 1, \qquad (i,j)\in E. \tag{3.3}$$

This model can be solved as a perfect matching problem on the graph (T, E_T), where $E_T = \{(i,j) : i,j \in T, i < j\}$ and the matching cost d_{ij} associated with $(i,j) \in E_T$ is the length of a shortest chain between i and j. The matching problem is solvable in $O(|T|^3)$ time (Lawler [26]). However, lower complexity algorithms that exploit data structures are also available (see, e.g., Galil, Micali, and Gabow [18] and Derigs and Metz [9]).

The paper by Edmonds and Johnson [13] provides the first mathematical programming treatment of the CPP. It proposes a compact formulation of the problem which does away with the modulo conditions. In what follows, $S \subset V$ is said to be odd if it is a set consisting of an odd number of odd-degree vertices. Let $E(S) = \{(i,j) : i \in S, j \in V\backslash S$, or $i \in V\backslash S, j \in S\}$. Using the same x_{ij} variables as in CPP1, the problem can be formulated as

$$\text{(CPP2)} \qquad \text{minimize} \sum_{(i,j)\in E} c_{ij}x_{ij} \tag{3.4}$$

$$\text{s.t.} \qquad \sum_{(i,j)\in E(S)} x_{ij} \geq 1, \quad S \subset V, S \text{ is odd}, \tag{3.5}$$

$$x_{ij} \geq 0, \quad (i,j)\in E, \tag{3.6}$$

$$x_{ij} \text{ integer}, \quad (i,j)\in E. \tag{3.7}$$

In this formulation, constraints (3.5) are referred to as *blossom inequalities* (or *odd-cut* inequalities). They state that in order to achieve evenness, at least one copy of an edge incident to an odd set S must be part of the augmented graph. Edmonds and Johnson prove that the polyhedron of solutions to (3.5) and (3.6) is equal to the convex hull of solutions of the problem and that blossom inequalities can be separated in polynomial time. They solve the problem through an adaptation of the blossom algorithm for matching problems (Edmonds [12]). However, given the efficiency of today's integer linear programming solvers, the use of a specialized algorithm for the matching problem may be less justified.

Once a Eulerian graph has been obtained, determining a Eulerian cycle on it can be achieved by applying Hierholzer's algorithm [23], for example (Edmonds and Johnson [13] described it under the name "end-pairing algorithm"):

Step 1. Starting from an arbitrary vertex v_1, construct a cycle $(v_1,\ldots,v,v_2,\ldots,v_1)$ by iteratively traversing untraversed edges until v_1 is reached.

Step 2. If all edges have been traversed, stop.

Step 3. Construct a second cycle starting from an edge (v,u_1) incident to the first cycle. Merge the two cycles $(v_1,\ldots,v,v_2,\ldots,v_1)$ and $(v,u_1,\ldots,u_2,v)$ into the single cycle $(v_1,\ldots,v,u_1,\ldots,u_2,v,v_2,\ldots,v_1)$, and go to Step 2.

The complexity of the Hierholzer algorithm is $O(|E|)$. An alternative algorithm was later proposed by Fleury [17]:

Step 1. Consider a vertex v_i. Starting with v_i, traverse an edge (v_i,v_j) that is not a bridge (an edge whose removal disconnects the graph), unless it is an end-edge. Remove (v_i,v_j) from the graph.

Step 2. If all edges have been removed, stop.

Step 3. Set $v_i := v_j$, and go to Step 1.

The main difficulty in Fleury's algorithm is to determine whether an edge is a bridge. If the Tarjan [31] $O(|E|)$ algorithm is used to this end, then this algorithm can run in $O(|E|^2)$ time. Fleury was apparently unaware of Hierholzer's more efficient algorithm. In his paper, he wrote, "Nobody has yet shown how to determine, without trial and error, a continuous traversal without repetitions" (Fleury [17, pp. 257–258], author's translation). Other graph traversal algorithms are provided in Fleischner [16].

3.3 ▪ Variants

The CPP variants are mostly inspired from their counterparts for the TSP and are typically NP-hard.

3.3.1 ▪ The generalized CPP

In the Generalized CPP, introduced by Dror and Haouari [10], the edge set is partitioned into $E = \{E_1,\ldots,E_p\}$. The aim is to determine a least-cost cycle containing at least one edge from each element of the partition. The authors do not actually solve this problem, but it can readily be transformed into a generalized TSP, and then into an asymmetric TSP using the technique proposed by Blais and Laporte [4]. This is achieved by first replacing each edge by two opposite arcs, one for each traversal direction, and then replacing each such arc by a vertex. All vertices of the transformed graph are then connected by arcs whose costs are those of shortest paths on the original graph. This gives rise to an asymmetric Generalized TSP which is then transformed into an asymmetric TSP over the same number of vertices, as suggested by Noon and Bean [29].

3.3.2 ▪ The cumulative CPP

The *Cumulative CPP* (CCPP) is rooted in the Cumulative TSP which is also known as the Traveling Repairman Problem (Afrati et al. [1]), the Deliveryman Problem (Lucena [27], Fischetti, Laporte, and Martello [15]), and the Minimum Latency Problem (Blum et al. [5]). In the Cumulative TSP, a salesman based at a depot must complete a Hamiltonian tour over a set of customers so as to minimize the sum of arrival times at the customer

locations. The CPP counterpart of this problem for arc routing was introduced by van Omme [32]. The problem is defined on an undirected graph, and one vertex is designated as the depot. The aim is to determine an Eulerian cycle starting and ending at the depot and minimizing the sum of service completions of all edges, which implies that if an edge is traversed several times, it is advantageous to service it during the first traversal. Note that in this definition, the return time to the depot is not included in the objective function whenever some deadheading takes place after servicing the last edge. This problem is strongly NP-hard and is reducible to a version of the Cumulative TSP. However, when the problem is defined on a linear or on a circular graph, it can be solved in $O(n^2)$ time. The CCPP arises naturally in several applications such as snow removal.

The author has developed eight models which were theoretically and empirically compared. The best of these, which we now describe, was retained to conduct the final computational experiments. As for the CPP, the cumulative problem is defined on an undirected graph $G=(V,E)$, where $|V|=n$ and $|E|=m$. This graph is expanded into an auxiliary mixed graph $\bar{G}=(\bar{V},\bar{E}\cup\bar{A})$. In $\bar{G}$, the edges of E are "exploded," i.e., represented as if they were mutually disjoint. Let $\bar{E}$ be this set of exploded edges, and note that the number of vertices incident to the edges of $\bar{E}$ is equal to $2n$. Let $\tilde{V}$ be the set of these vertices, and let $\bar{V}=\tilde{V}\cup\{s,t\}$, where s is a starting depot and t is an ending depot. To define $\bar{A}$, create arcs between the vertices of $\bar{V}$ incident to different edges, from s to the two vertices of $\bar{V}$ corresponding to the depot in G, and from the vertices of $\bar{V}$ to t. This construction is illustrated in the small graph of Figure 3.1.

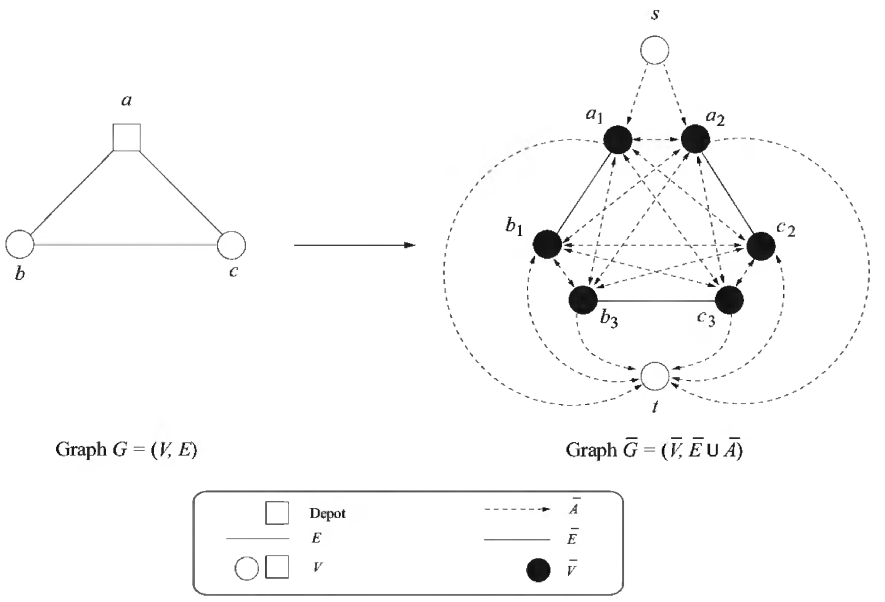


Figure 3.1. *Construction of the graph $\bar{G}$, from van Omme [32].*

To define the cost structure, denote by $*v$ the vertex of V corresponding to vertex $v\in\bar{V}$. Thus, in Figure 3.1 $*a_1=a$. Given $v\in\bar{V}$, the vertex $\bar{v}\in\bar{V}$ is the other vertex of edge $(v,\bar{v})\in\bar{E}$. For example, in the graph of Figure 3.1, $\bar{a}_1=b_1$. Let c_{uv} be the cost of edge $(u,v)\in E$, and let d_{uv} be the cost of a shortest chain between u and v in G. For any edge $(i,j)\in\bar{E}$, let $\bar{c}_{ij}=d_{*i,*j}+c_{*j,*\bar{j}}$.

The variables of the model are defined as follows:

(1) If $i, j \in \tilde{V}$, let

$$y_{ij}^k = \begin{cases} 1 & \text{if arc } (i,j) \text{ or arc } (j,i) \text{ is traversed in position } k \text{ (which} \\ & \text{means that edge } (*j, *\bar{j}) \text{ is visited in position } k \text{ in } G), \\ 0 & \text{otherwise.} \end{cases}$$

(2) Otherwise, let

$$y_{sv}^1 = \begin{cases} 1 & \text{if } v \in \tilde{V} \text{ is the first vertex visited after } s \text{ in } \bar{G}, \\ 0 & \text{otherwise,} \end{cases}$$

$$y_{vt}^{m+1} = \begin{cases} 1 & \text{if } v \in \tilde{V} \text{ is the last vertex visited before } t \text{ in } \bar{G}, \\ 0 & \text{otherwise.} \end{cases}$$

The model is then

$$\text{(CCPP)} \qquad \text{minimize} \sum_{k=1}^{m} \sum_{(i,j)\in\bar{A}} (m-k+1)\bar{c}_{ij} y_{ij}^k$$

$$\text{(3.8)} \qquad \text{s.t.} \quad y_{sv}^1 + \sum_{k=2}^{m} \Big(\sum_{(v,i)\in\bar{A}} y_{vi}^k + \sum_{(j,v)\in\bar{A}} y_{jv}^k \Big) + j_{vt}^{m+1} = 1, \qquad v \in \bar{V},$$

$$\text{(3.9)} \qquad \sum_{(i,j)\in\bar{A}} y_{ij}^k = 1, \qquad k \in \{1, \ldots, m+1\},$$

$$\text{(3.10)} \qquad \sum_{(p,i)\in\bar{A}} y_{pi}^k = \sum_{(i,q)\in\bar{A}} y_{iq}^{k+1}, \qquad i \in \bar{V}, k \in \{1, \ldots, m\},$$

$$\text{(3.11)} \qquad y_{ij}^k = 0 \text{ or } 1, \qquad i, j \in \bar{V}, k \in \{1, \ldots, m\}.$$

In this formulation, constraints (3.8) force every edge of E to be serviced. Constraints (3.9) state that for each position k a shortest path must be visited. Constraints (3.10) are flow conservations constraints which guarantee the creation of a tour.

This model contains $O(m^3)$ variables and $O(m^2)$ constraints. Even on relatively small graphs, these numbers can be quite large. However, van Omme notes that most of the variables are equal to zero in an optimal solution. He therefore proposes several preprocessing operations to eliminate many of the variables.

The problem was solved by means of an implicit enumeration heuristic. Several versions of a branch-and-cut algorithm were also developed and compared. Among six families of cuts developed, three were retained: odd set inequalities, even set inequalities, and generalized co-circuit inequalities. Two branch-and-price-and-cut algorithms were also developed and applied to a streamlined version of the CCPP model (with a reduced number of variables), including this time five families of cuts.

The algorithms were tested on five graphs with $10 \leq n \leq 20$ and $30 \leq m \leq 45$. Results indicate that the branch-and-price-and-cut algorithm is two to four times faster than the branch-and-cut algorithm, and two to 133 times faster than CPLEX applied to the streamlined CCPP model.

3.3.3 ▪ The hierarchical CPP

In some applications such as snow plowing (Alfa and Liu [2], Haslam and Wright [22]), garbage collection (Bodin and Kursh [6]) and flame cutting (Manber and Israni [28]), constraints may be imposed on the order in which some *clusters* of edges (or arcs) must be

visited. Here we restrict our attention to undirected graphs. This problem, called the *Hierarchical CPP* (HCPP), was introduced by Dror, Stern, and Trudeau [11]. It is defined on a graph $G=(V,E)$, where V includes a depot d. The edges are partitioned into $\{E_1,\ldots,E_p\}$, where p is the number of clusters. An order relation $\prec$ is imposed on the elements of the partition. This means that if $E_h \prec E_k$, then all edges of E_h must be serviced before any edge of E_k. As shown by Dror, Stern, and Trudeau, the HCPP is NP-hard. However, when the subgraphs $G_k=(V_k,E_k)$ induced by the edges of E_k are connected for each k and a total order relation $E_1 \prec \cdots \prec E_p$ is prespecified, then the problem can be solved in polynomial time through the construction of an auxiliary graph $\bar{G}$. Using such a transformation, Dror, Stern, and Trudeau developed an $O(p|V|^5)$ algorithm. This complexity was later reduced to $O(p^3|V|^3)$ by Ghiani and Improta [19] and to $O(p|V|^4)$ by Korteweg and Volgenant [24].

3.3.3.1 ▪ The Dror, Stern, and Trudeau algorithm

Dror, Stern, and Trudeau [11] proved a necessary and sufficient condition for a feasible HCPP solution to exist. Let G_k be the subgraph induced by the edges of E_k, and let F_k be the subgraph induced by the edges of $E_1 \cup \cdots \cup E_k$. Then there exists a feasible HCPP solution if and only if F_k is connected for $k=1,\ldots,p$.

The graph $\bar{G}$ is constructed as follows. Let $\bar{V}_1=\bar{V}_{p+1}=\{d\}$. For $k=2,\ldots,p$, let $\bar{V}_k$ be the set of *entry vertices* of G_k, i.e., the vertices of V_k incident to an edge of $E_1 \cup \cdots \cup E_{k-1}$.

Step 1. Construct a $(p+1)$-bipartite graph $\bar{G}$ with layers of vertices $\bar{V}_1,\ldots,\bar{V}_{p+1}$. Define an arc (u,v) from every $u \in \bar{V}_k$ and $v \in \bar{V}_{k+1}$ for $k=1,\ldots,p$. The cost $\bar{c}_{uv}$ of arc (u,v) is equal to that of a shortest Eulerian path from u to v in F_k.

Step 2. Construct a shortest path from the copy of the depot in $\bar{V}_1$ to that of the depot in $\bar{V}_{p+1}$.

Step 3. Connect the paths that correspond to the edges of G belonging to the shortest path in $\bar{G}$, which yields an optimal HCPP solution in G.

To illustrate, consider the graph G depicted in Figure 3.2, with $d=1$, $E_1=\{(1,2),(2,3),(2,4)\}$, $E_2=\{(3,5),(4,6),(4,7),(5,6),(6,7)\}$, and $E_3=\{(1,8),(2,8)\}$, with $E_1 \prec E_2 \prec E_3$. The graph $\bar{G}$ corresponding to G (Figure 3.2) is depicted in Figure 3.3. The optimal solution is given by the shortest path $(1,3,2,1)$ of cost 30 in $\bar{G}$, corresponding to the optimal solution $(1,2,4,2,3,5,6,7,4,6,4,2,8,1)$ in G.

The $O(p|V|^5)$ complexity of the Dror, Stern, and Trudeau algorithm is determined by Step 1, which requires the solution of $O(p|V|^2)$ matching problems of complexity $O(|V|^3)$. Korteweg and Volgenant [24] reduce this complexity to $O(p|V|^4)$ by exploiting the relationship between the solutions of successive Eulerian path computations.

3.3.3.2 ▪ The Ghiani and Improta algorithm

The Ghiani and Improta [19] algorithm solves a single matching problem on an auxiliary graph G^* containing $O(p|V|)$ vertices, and the authors show that this operation is the most complex step of their procedure. Hence, the overall time complexity of their algorithm is $O(p^3|V|^3)$, which can yield significant savings with respect to the previous two algorithms since in practice p is usually small compared with $|V|$.

We now outline this algorithm. In a solution, after the edges of $E_{k-1}(k=2,\ldots,p)$ have been serviced, one must traverse a path $\mathscr{P}_k$ in F_{k-1} linking V_{k-1} to V_k in order to

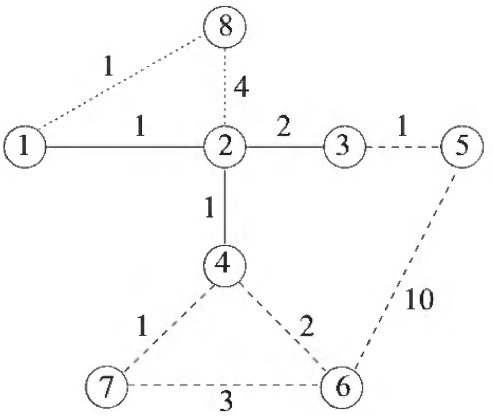


Figure 3.2. *Graph G, from Dror, Stern, and Trudeau [11].*

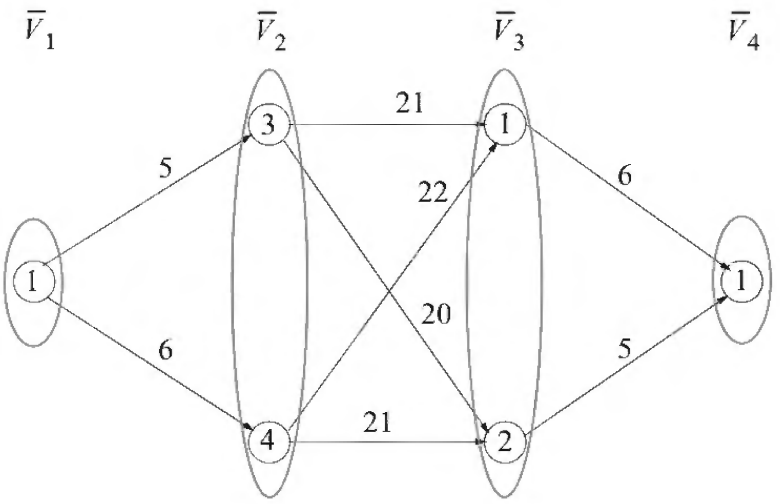


Figure 3.3. *Graph $\bar{G}$ corresponding to graph G (Figure 3.2), from Dror, Stern, and Trudeau [11].*

serve the first edge of E_k. The path $\mathscr{P}_k$ contains a subpath $(v''_{k-1},\ldots,v'_k)$, where $v''_{k-1} \in V''_{k-1}$, the set of vertices of V_{k-1} incident to at least one edge of $E_1 \cup \cdots \cup E_{k-2} \cup E_k$, and $v'_{k-1} \in V'_{k-1}$, the set of vertices of V_k incident to at least one edge of $E_1 \cup \cdots \cup E_{k-1}$. Let $P = P^L \cup P_1 \cup \cdots \cup P_p$, where P^L is a set of paths $\{(v''_1,\ldots,v''_2),\ldots,(v''_p,\ldots,d)\}$ linking V_k to V_{k+1} in $F_k (k = 1,\ldots,p-1)$ and V_p to d in F_p; P_k is a set of paths in F_k connecting two vertices in $V_k (k = 1,\ldots,p)$.

The graph G^* contains $p+2$ layers $V_0^* = \{d\}$, $V_1^*,\ldots,V_p^*$, $V_{p+1}^* = \{d\}$, where V_k^* contains as many copies of each vertex $v \in V_i$ as the maximum number of paths in $P^L \cup P_k$ incident to v. Each copy v_k of a given vertex $v \in V_k$ is connected to its copies by a zero cost edge and to the copies in V_k^* of the other vertices $v' \in V_i$ by an edge of cost equal to that of a shortest path from v to v' in F_k. In addition, if $v \in V'_k$, v_k is connected to the copies in V_{k-1}^* of vertices v' in V_{k-1}^* by an edge of cost equal to that of a shortest path from v to v' in F_{k-1}, plus (if $k \neq 1$ and $k \neq p+1$) a large constant M. Finally, if $v' \in V''_k$,

v_k is linked to the copies in V^*_{k+1} of the vertices $v' \in V'_{k+1}$ by an edge of cost equal to that of shortest path from v to v' in F_k, plus (if $k \neq 0$ and $k \neq p$) a large constant M. The graph G^* corresponding to G (Figure 3.1) is depicted in Figure 3.4.

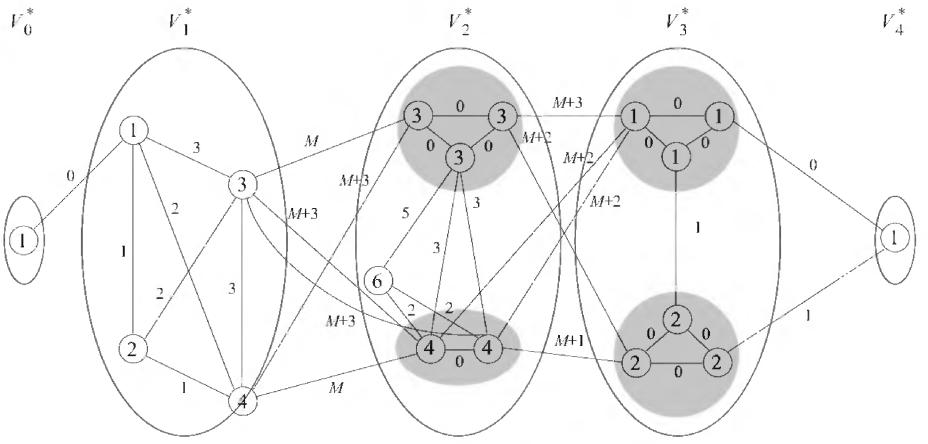


Figure 3.4. *Graph G^* corresponding to graph G (Figure 3.1), from Ghiani and Improta [19].*

The algorithm is then as follows:

Step 1. For each subgraph $G_k (k = 1, \ldots, p)$, determine V'_k and V''_k.

Step 2. Construct the auxiliary graph G^* as described above.

Step 3. Compute a minimum cost perfect matching on G^*.

Step 4. Introduce in G the set P of shortest paths corresponding to the solution found in Step 3. Let G' be the resulting graph.

Step 5. Compute a cycle on G' by means of the end-pairing algorithm.

The complexity of this algorithm is determined as follows. Step 1 requires $O(p|V|^2)$ operations. Step 2 requires $O(p|V|^3)$ operations since it involves the computation of p shortest paths on graphs having at most $|V|$ vertices. The solution of the matching problem in Step 3 requires $O(p^3|V|^3)$ operations since each of the p_2 layers of G^* has at most $3|V|$ vertices. Finally, Step 5 can be executed in $O(|E|)$ time. The overall complexity is therefore that of Step 3.

3.3.3.3 ▪ Transformation into a Rural Postman Problem

Cabral et al. [7] consider a version of the HCPP in which the vertex set contains a depot d, the edge set is partitioned into $E = \{E_1, \ldots, E_p\}$, and a total relation $\prec$ is imposed on the elements of the partition, but these may not be connected. Each edge e has three associated lengths (traversal times): t^3_e is the time of servicing e, t^2_e is the time of traversing e if it has not yet been serviced, and t^1_e is the time of traversing e if it has already been serviced. Typically $t^3_e \geq t^2_e \geq t^1_e$, but there exist applications, such as snow removal, where $t^2_e > t^3_e$ (often $t^2_e = \infty$), in which case feasibility is not guaranteed. The authors consider two different objectives. The first is a *makespan* objective which consists of minimizing the return time at the depot once all edges have been traversed. The second is a *hierarchical* objective in which the aim is to minimize $\sum_{k=1}^{p} M_k T_k$, where T_k is the time at which all

edges of E_k have been serviced and $M_1 \gg \cdots \gg M_p = 1$, which means that the edges of E_k must be serviced before those of $E_{k+1} (k = 1, \ldots, p-1)$. In practice, the notation "$\gg$" means that in any feasible solution, the inequality $M_k T_k > \sum_{h=k+1}^{p} M_h T_h$ must be satisfied for $k = 1, \ldots, p-1$. This inequality is satisfied if it holds when T_k is replaced by a lower bound and each T_h is replaced by an upper bound.

The authors show that for either objective, the HCPP can be cast as a Rural Postman Problem (RPP) (see Chapter 5) through a graph transformation, the advantage being that good algorithms are readily available for the RPP (see, e.g., Ghiani and Laporte [20]). The transformation works as follows. Let $\alpha(e)$ denote the cluster of edge $e \in E$. Define the modified edge lengths

$$\tilde{t}_e^k = \begin{cases} t_e^1 & \text{if } \alpha(e) \leq k, \\ t_e^2 & \text{if } \alpha(e) > k, \end{cases}$$

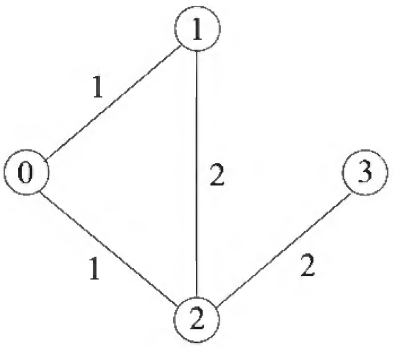
and let s_{ij}^k be the length of a shortest path in G between vertices i to j with respect to the modified edge lengths. The problem is solved on a $(p+2)$-layered multigraph $\tilde{G} = (\tilde{V}, \tilde{E})$. The vertex set $\tilde{V}$ is partitioned into $\tilde{V} = \{\tilde{V}_0, \ldots, \tilde{V}_{p+1}\}$, where $\tilde{V}_0 = \tilde{V}_{p+1} = \{d\}$, and $\tilde{V}_h = V_h$. For any two vertices i and j belonging to the same layer k $(k = 1, \ldots, p)$, define a nonrequired edge of length $\sum_{h=1}^{k} M_h s_{ij}^k$; if $(i, j) \in E_k$, also define a required edge of length $\sum_{h=1}^{k} M_h t_e^3$ between i and j. For any two vertices i and j with $i \in \tilde{V}_{k-1}$ and $j \in \tilde{V}_k$ $(k = 1, \ldots, p)$, define a nonrequired edge of length $\sum_{h=1}^{k-1} M_h s_{ij}^k + M$, where $M \gg M_1$. Finally, define a nonrequired edge of length M between the vertices of $\tilde{V}_p$ and the vertex of $\tilde{V}_{p+1}$, as well as a required edge of cost 0 linking the two copies of the depot. Figure 3.5 depicts this graph construction on a small example.

Solving the HCPP can become rather time consuming as the instance size grows. Therefore, in order to obtain good solutions quickly on larger instances, the authors propose two heuristics. The best one is a decomposition heuristic which works as follows. For $k = 1, \ldots, p$, using the RPP transformation, solve the problem on the subgraph induced by the edges of $E_1 \cup \cdots \cup E_k$ and declare all traversed edges already serviced for the following iterations. The authors performed computational experiments on randomly generated instances with $p = 3$ and $30 \leq |V| \leq 150$. They could easily solve most instances within 15 minutes with the makespan objective, but they could not solve any instance with $|V| \geq 60$ within the same time under the hierarchical objective. However, they showed that using either objective does not make much difference as far as the objective function value is concerned. They also showed that the decomposition heuristic is quick and yields very small optimality gaps.

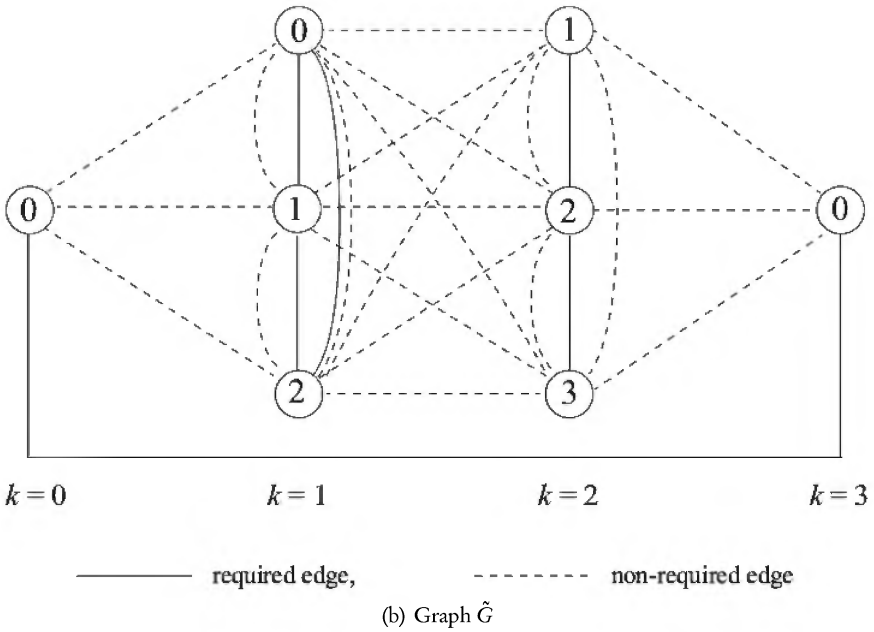
3.3.4 ▪ The CPP with time windows

The undirected CPP with time windows is defined on a graph $G = (V, E)$, where a time window $[\ell_e, u_e]$ is imposed on the start and completion of service of every edge $e \in E$. This NP-hard problem was introduced by Aminu and Eglese [3]. It arises naturally in several arc routing applications related to street sweeping and snow removal, for example. In several cities, parking interdictions are enforced on some streets during certain periods in order to allow such operations to take place. These no-parking periods define the time windows.

The authors have developed two exact *constraint programming* (CP) algorithms for the CPP with time windows. The first one operates directly on G and assigns a position



(a) Graph G showing edge priorities



(b) Graph $\tilde{G}$

Figure 3.5. *Transformation of the HCPP into an RPP, from Cabral et al. [7].*

to each edge in the solution. The CP mechanism operates in a branch-and-bound fashion and gradually restricts the domains of the position variables. The second CP implementation is carried out on a transformed graph $\bar{G}$ whose vertex set $\bar{V}$ is the union of $|E|$ clusters of two vertices, one for each of the two end vertices of every $e \in E$. This graph transformation is a simplification of the procedure proposed by Pearn, Assad, and Golden [30], but it replaces each edge with two vertices instead of three. Arcs are then created between the vertices of $\bar{G}$ to represent shortest paths in G, which correspond to costs. Solving a clustered TSP on $\bar{G}$ (see, e.g., Laporte and Palekar [25]) solves the CPP with time windows. The authors have compared two CP implementations on graphs containing between seven and 69 vertices and have shown that the second CP implementation is superior to the first one.

Bibliography

[1] F. AFRATI, S. COSMADAKIS, S. PAPADIMITRIOU, S. PAPAGEORGIOU, AND N. PAPAKOSTANTINOU, *The complexity of the traveling repairmen problem*, RAIRO Informatique Théorique et Applications, 20 (1986), pp. 79–87.

[2] A. S. ALFA AND D. Q. LIU, *Postman routing problem in a hierarchical network*, Engineering Optimization, 14 (1988), pp. 127–138.

[3] U. F. AMINU AND R. W. EGLESE, *A constraint programming approach to the Chinese postman problem with time windows*, Computers & Operations Research, 33 (2006), pp. 3423–3431.

[4] M. BLAIS AND G. LAPORTE, *Exact solution of the generalized routing problem through graph transformations*, The Journal of the Operational Research Society, 54 (2003), pp. 906–910.

[5] A. BLUM, P. CHALSANI, D. COPPERSMITH, B. PULLEYBLANK, P. RAGHAVAN, AND M. SUDAN, *The minimum latency problem*, in STOC'94, Proceedings of the 26th Annual ACM Symposium on the Theory of Computing, ACM, New York, 1994, pp. 163–171.

[6] L. D. BODIN AND S. J. KURSH, *A computer-assisted system for the routing and scheduling of street sweepers*, Operations Research, 26 (1978), pp. 525–537.

[7] E. A. CABRAL, M. GENDREAU, G. GHIANI, AND G. LAPORTE, *Solving the hierarchical Chinese postman problem as a rural postman problem*, European Journal of Operational Research, 155 (2004), pp. 44–50.

[8] N. CHRISTOFIDES, *Worst-case analysis of a new heuristic for the travelling salesman problem*, Tech. Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA, 1976.

[9] U. DERIGS AND A. METZ, *Solving (large scale) matching problems*, Mathematical Programming, 50 (1991), pp. 113–121.

[10] M. DROR AND M. HAOUARI, *Generalized Steiner problems and other variants*, Journal of Combinatorial Optimization, 4 (2000), pp. 415–436.

[11] M. DROR, H. STERN, AND P. TRUDEAU, *Postman tour on a graph with precedence relation on arcs*, Networks, 17 (1987), pp. 283–294.

[12] J. EDMONDS, *Paths, trees and flowers*, Canadian Journal of Mathematics, 17 (1965), pp. 449–467.

[13] J. EDMONDS AND E. L. JOHNSON, *Matching, Euler tours and the Chinese postman*, Mathematical Programming, 5 (1973), pp. 88–124.

[14] L. EULER, *Solutio problematis ad geometriam situs pertinentis*, Commentarii Academiae Scientarum Petropolitanae, 8 (1736), pp. 128–140.

[15] M. FISCHETTI, G. LAPORTE, AND S. MARTELLO, *The delivery man problem and cumulative matroids*, Operations Research, 41 (1993), pp. 1055–1064.

[16] M. FLEISCHNER, *Eulerian Graphs and Related Topics*, Annals of Discrete Mathematics 2, North-Holland, Amsterdam, 1991.

[17] M. FLEURY, *Deux problèmes de géométrie de situation*, Journal de Mathématiques Élémentaires, 2 (1883), pp. 257–261.

[18] Z. GALIL, S. MICALI, AND H. GABOW, *An O(EV* log *V) algorithm for finding a maximal weighted matching in general graphs*, SIAM Journal on Computing, 15 (1986), pp. 120–130.

[19] G. GHIANI AND G. IMPROTA, *An algorithm for the hierarchical Chinese postman problem*, Operations Research Letters, 26 (2000), pp. 27–32.

[20] G. GHIANI AND G. LAPORTE, *A branch-and-cut algorithm for the undirected rural postman problem*, Mathematical Programming, 87 (2000), pp. 467–481.

[21] M. GUAN, *Graphic programming using odd and even points*, Chinese Mathematics, 1 (1962), pp. 273–277.

[22] E. HASLAM AND J. R. WRIGHT, *Application of routing technologies to rural snow and ice control*, Transportation Research Records, 1304 (1991), pp. 202–211.

[23] C. HIERHOLZER, *Uber die Möglichkeit, einen Linienzug ohne Wiederholung und ohne Unterbrechnung zu umfahren*, Mathematische Annalen, 6 (1873), pp. 30–32.

[24] P. KORTEWEG AND T. VOLGENANT, *On the hierarchical Chinese postman problem with linear ordered classes*, European Journal of Operational Research, 169 (2006), pp. 41–52.

[25] G. LAPORTE AND U. PALEKAR, *Some applications of the clustered traveling salesman problem*, The Journal of the Operational Research Society, 53 (2002), pp. 972–976.

[26] E. L. LAWLER, *Combinatorial Optimization: Networks and Matroids*, Holt, Reinhart & Winston, New York, 1976.

[27] A. LUCENA, *Time-dependent traveling salesman problem—the deliveryman case*, Networks, 20 (1990), pp. 753–763.

[28] U. MANBER AND S. ISRANI, *Pierce point minimization and optimal torch path determination in flame cutting*, Journal of Manufacturing Systems, 3 (1984), pp. 81–89.

[29] C. E. NOON AND J. C. BEAN, *An efficient transformation of the generalized traveling salesman problem*, INFOR, 31 (1993), pp. 39–44.

[30] W.-L. PEARN, A. A. ASSAD, AND B. L. GOLDEN, *Transforming arc routing into node routing problems*, Computers & Operations Research, 14 (1987), pp. 285–288.

[31] R. E. TARJAN, *A note on finding the bridges of a graph*, Information Processing Letters, 1 (1974), pp. 160–161.

[32] N. VAN OMME, *Le problème du postier chinois cumulatif*, Ph.D. thesis, Département de mathématiques et de génie industriel, Polytechnique Montréal, Montréal, Canada, 2011.

Chapter 4

The Chinese Postman Problem on Directed, Mixed, and Windy Graphs

Ángel Corberán
Isaac Plana
José María Sanchis

4.1 ▪ Introduction

Let $G = (V, E, A)$ be a mixed graph, consisting of a set of vertices V, a set of (undirected) edges E, and a set of (directed) arcs A. For simplicity, the term *link* will be used to refer to both edges and arcs indistinctly. Note that if $E = \emptyset$, G is a directed graph (all the links are arcs that must be traversed in a specified direction), while if $A = \emptyset$, it is an undirected graph (all the links are edges that can be traversed in both directions at the same cost). Usually, the links in undirected, directed, and mixed graphs have a nonnegative cost associated with them. If G is an undirected graph and there are two costs associated with each edge, representing the cost of traversing it in each possible direction, G is called a windy graph. Note that, since any arc (i, j) with cost c_{ij} can be modeled as an edge joining i and j with the same cost from i to j and infinite cost in the opposite direction, a mixed graph can be considered as a special case of a windy graph.

The *Chinese Postman Problem* (CPP) consists of finding a minimum cost tour (a closed walk) traversing every link of G at least once. This well-known problem (Guan [20]) was shown to be solvable in polynomial time for undirected and directed graphs. If the CPP is defined over a mixed graph $G = (V, E, A)$, the problem of traversing all the edges and arcs in G at total minimum cost is called the *Mixed Chinese Postman Problem* (MCPP). Papadimitriou [32] showed the NP-hardiness of the MCPP. If $G = (V, E)$ is a windy graph, the problem of traversing all the edges in G at total minimum cost is called the *Windy Postman Problem* (WPP). Since the MCPP is a particular case of the WPP, it is easy to see that the WPP is also an NP-hard problem (Brucker [2], Guan [21]).

If an (undirected, directed, mixed, or windy) graph G is *Eulerian*, there is a tour passing through each link of G exactly once. Obviously, this tour is optimal. Furthermore, this (Eulerian) tour can be easily computed (see, for instance, Eiselt, Gendreau, and Laporte [14]), and hence graph G can itself be considered as the CPP solution. If an (undirected, directed, mixed, or windy) graph G is not Eulerian, the CPP can be formulated as the

problem of finding a set of copies of links at minimum cost, such that when added to G, an Eulerian graph is obtained. As before, the "augmented" graph formed by G, plus the added copies of the links, can be considered as the solution of the problem.

For the undirected CPP, an Eulerian graph can be obtained from G by adding edges to match odd-degree vertices (Christofides [4], Edmonds and Johnson [13]). This can be done by finding a minimum cost perfect matching (see Chapter 3).

In this chapter we study the directed CPP and the two NP-hard versions of the CPP, the MCPP and the WPP, as well as some variants.

4.2 ▪ The directed Chinese Postman Problem

Given a strongly connected directed graph $G = (V,A)$, with nonnegative costs c_{ij} for each $(i,j) \in A$, the *Directed Chinese Postman Problem* (DCPP) is the problem of finding a minimum cost tour passing through each arc in A at least once.

In order to formulate the problem we introduce the following notation. For each vertex i, let $\delta^+(i)$ denote the set of arcs leaving i, $\delta^-(i)$ the set of arcs entering i, and $s_i = |\delta^-(i)| - |\delta^+(i)|$. For each arc $(i,j) \in A$, let x_{ij} be the number of copies of each arc that must be added to G in order to obtain an Eulerian graph. Edmonds and Johnson [13] showed that the DCPP can be formulated as the following linear program (LP):

$$\text{(4.1)} \quad \text{(DCPP)} \quad \min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

$$\text{(4.2)} \quad \sum_{(i,j)\in\delta^+(i)} x_{ij} - \sum_{(j,i)\in\delta^-(i)} x_{ji} = s_i \qquad \forall i \in V,$$

$$\text{(4.3)} \quad x_{ij} \geq 0 \qquad \forall (i,j) \in A.$$

This corresponds to a minimum cost flow problem on G with supplies s_i for vertices i with $s_i > 0$, and demands $-s_i$ for vertices i with $s_i < 0$ (Liebling [26], Edmonds and Johnson [13]).

4.3 ▪ The mixed Chinese Postman Problem

In this section we give a formal definition of the MCPP, present some notation and two different formulations, and discuss its heuristic and exact solution.

Given a strongly connected mixed graph $G = (V,E,A)$ with a vertex set V, an edge set E, an arc set A, and a nonnegative cost c_e for each $e \in E \cup A$, the MCPP consists of finding a minimum-cost tour passing through each link $e \in E \cup A$ at least once.

Given $S \subset V$, $\delta^+(S) = \{(i,j) \in A : i \in S, j \in V \setminus S\}$, $\delta^-(S) = \{(i,j) \in A : i \in V \setminus S, j \in S\}$, $\delta(S)$ denotes the set of edges with exactly one endpoint in S, and $\delta^*(S) = \delta(S) \cup \delta^+(S) \cup \delta^-(S)$. Given a vertex i, d_i^- (indegree) denotes the number of arcs entering i, d_i^+ (outdegree) is the number of arcs leaving i, and d_i (degree) is the number of links incident with i. Note that $d_i = |\delta^*(\{i\})|$. A mixed graph $G = (V,E,A)$ is called *even* if all its vertices have even degree, it is called *symmetric* if $d_i^- = d_i^+$ for each vertex i, and it is said to be *balanced* if, given any subset S of vertices, the difference between the number of arcs directed from S to $V \setminus S$, $|\delta^+(S)|$, and the number of arcs directed from $V \setminus S$ to S, $|\delta^-(S)|$, is no greater than the number of (undirected) edges joining S and $V \setminus S$, $|\delta(S)|$.

It is well known that a mixed graph G is Eulerian if and only if G is even and balanced (Ford and Fulkerson [16]). Notice that if G is even and symmetric, then G is also balanced

(and Eulerian). Moreover, if G is even, the MCPP can be solved exactly in polynomial time (Edmonds and Johnson [13]). The procedure, as described by Minieka [27], works as follows:

Even MCPP Algorithm

(1) Given an even and strongly connected mixed graph, $G = (V,E,A)$, let A_1 be the set of arcs obtained by arbitrarily assigning a direction to the edges in E and with the same costs. Compute $s_i = d_i^- - d_i^+$ for each vertex i in the directed graph $(V, A \cup A_1)$. A vertex i with $s_i > 0$ ($s_i < 0$) will be considered as a source (sink) with supply (demand) s_i ($-s_i$). Note that as G is an even graph, all supplies and demands are even numbers (zero is considered an even number).

(2) Let A_2 be the set of arcs in the opposite direction to those in A_1 and with the costs of the corresponding edges, and let A_3 be a set of arcs parallel to those of A_2 at zero cost.

(3) To satisfy the demands s_i of all the vertices, solve a minimum cost flow problem in the graph $(V, A \cup A_1 \cup A_2 \cup A_3)$, in which each arc in $A \cup A_1 \cup A_2$ has infinite capacity and each arc in A_3 has capacity 2. Let x_{ij} be the optimal flow.

(4) For each arc (i,j) in A_3 do: If $x_{ij} = 2$, then orient the corresponding edge in G from i to j (the direction, from j to i, assigned to the associated edge in step 1 was "wrong"); if $x_{ij} = 0$, then orient the corresponding edge in G from j to i (in this case, the orientation given in step 1 was "right"). Note that the case $x_{ij} = 1$ is impossible, as all flow values through arcs in A_3 are even numbers.

(5) Augment G by adding x_{ij} copies of each arc in $A \cup A_1 \cup A_2$. The resulting graph is even and symmetric.

4.3.1 ▪ Heuristic algorithms

When the mixed graph $G = (V,E,A)$ is not even, the above algorithm is the basis for two heuristics suggested by Edmonds and Johnson [13] and developed and improved by Frederickson [17]. Algorithm MIXED1 would be equivalent to first transforming G into an even graph and then applying the previous *Even MCPP Algorithm*. Specifically,

Algorithm MIXED1

(1 Evendegree) Let $G = (V,E,A)$ be a strongly connected mixed graph. Ignoring arc directions, solve a minimum-cost matching of odd-degree vertices and augment G by adding all links used for the matching solution. The resulting graph $G' = (V,E',A')$ is an even graph.

(2 Inoutdegree) With supplies and demands computed in graph (V,A'), solve a minimum cost flow problem in G' to obtain a symmetric graph $G'' = (V,E'',A'')$. (Frederickson points out that after this step some vertices in G'' may have odd degree.)

(3 Evenparity) Let V_O be the set of odd-degree nodes in G''. Identify cycles consisting of alternating paths in $A'' \setminus A$ and E'', with each path starting

and ending at vertices in V_0. The paths in $A'' \setminus A$ will be formed without considering the arc directions. As the cycles are covered, their arcs are either duplicated or deleted and their edges are directed, so the resulting graph becomes even, while maintaining the symmetry for each vertex.

Steps 1 and 2 above were first proposed by Edmonds and Johnson [13], who show that there is a minimum cost solution to the flow problem in step 2 which preserves the property that each vertex is of even degree. But, as noted by Frederickson [17], this argument merely proves the existence of such a solution. Step 3 above is a simple linear-time algorithm proposed by Frederickson for performimg this task.

Algorithm MIXED2 can be considered as the reverse version of MIXED1. It first solves a minimum-cost flow problem in G, with supplies and demands computed in graph (V,A), to obtain a symmetric graph (V,E',A'). In a second step, it solves the (undirected) CPP on each connected component of subgraph (V,E') to finally obtain an even and symmetric graph.

Frederickson [17] showed that both algorithms, MIXED1 and MIXED2, have a worst-case ratio of 2, but the *Mixed Algorithm*, which consists of applying both algorithms and selecting the best tour obtained, has a worst-case ratio of 5/3. Hertz and Mittaz [23] provide some illustrations and other details on the above heuristic procedures.

Twenty years later, Raghavachari and Veerasamy [34] proposed a modification to Frederickson's Mixed Algorithm with a better worst-case ratio of $3/2$:

Modified Mixed Algorithm

- Given $G = (V,E,A)$, solve a minimum-cost flow problem in G with supplies and demands computed in the graph (V,A). The resulting graph is (V,E',A'), where $E' \subseteq E$ are the edges of G that were not oriented (used) by the flow solution and $A' \supseteq A$ are the arcs that satisfy indegree equal to outdegree at each vertex.
- Before running Evendegree of MIXED1 algorithm, reset the costs of all arcs and edges in A' to 0, forcing Evendegree of MIXED1 to duplicate edges and arcs in A' whenever possible.
- Use the original costs for the rest of the MIXED1 algorithm.
- There are no changes in the MIXED2 algorithm.

In addition to its good theoretical behavior, the Modified Mixed Algorithm also shows good performance in practice, especially in graphs with a medium-high percentage of arcs, as was shown by Corberán, Martí, and Sanchis [6]. The following table, taken from [6], reports the percentage deviation from a lower bound (obtained with the cutting plane-algorithm described by Corberán, Romero, and Sanchis [11]) of the three constructive methods in three sets of randomly generated instances. Each set contains 75 instances of 50, 100, and 200 vertices, respectively, and a number of links randomly chosen between $2|V|$ and $3|V|$. Each set is divided into three groups of 25 instances according to the percentage of generated arcs: 30%, 50%, and 70%. The average number of links for each set is shown in the column *Links*.

The computational results, obtained in negligible computing times, show that Modified MIXED1 is significantly better than the original MIXED1 algorithm, except for those instances with a lower percentage of arcs. Moreover, its performance improves slightly when the percentage of arcs increases, while in the original MIXED1 the quality of the solutions clearly deteriorates as this percentage increases. On the other hand, MIXED2 presents the worst results, except for those instances with a large number of arcs.

Table 4.1. *Computational comparison of constructive heuristics for the MCPP.*

$\|V\|$	Links	Arcs	Mixed 1 Gap (%)	Mixed 2 Gap (%)	M. Mixed 1 Gap (%)
50	132	30%	1.93	16.63	3.18
		50%	7.10	20.41	3.15
		70%	11.25	11.18	1.84
100	257	30%	2.71	17.99	2.75
		50%	6.93	19.83	2.51
		70%	11.10	10.16	1.83
200	510	30%	3.14	18.92	3.39
		50%	7.62	19.67	2.62
		70%	11.15	9.43	2.07

A greedy randomized adaptive search procedure heuristic for the MCPP is also presented by Corberán, Martí, and Sanchis [6]. Other heuristics and improvement procedures for the MCPP are described in Pearn and Chou [33] and Yaoyuenyong, Charnsethikul, and Chankong [41].

4.3.2 ▪ Problem formulations

The MCPP, as well as all other arc routing problems defined on mixed graphs, can be formulated in two different ways, depending on whether only one or two variables associated with each edge of the graph are used. In the first case, the only variable x_e associated with edge e represents the number of times this edge is traversed (in any direction) by the solution. In the second case, two variables per edge are used, representing the number of times it is traversed in the corresponding direction. Both formulations have been widely used in the literature and have been compared from the theoretical and computational point of view in Corberán, Mota, and Sanchis [8].

4.3.2.1 ▪ Formulation F1

Formulation F1 is based on the necessary and sufficient condition given by Ford and Fulkerson [16] for a mixed graph to be Eulerian. Let x_e represent the number of times a link $e \in E \cup A$ appears in the tour. For a subset of links F, $x(F) = \sum_{e \in F} x_e$. Thus, the MCPP can be formulated as

$$\text{(4.4)} \quad \text{(MCPP-F1)} \quad \min \sum_{e \in E} c_e x_e + \sum_{(i,j) \in A} c_{ij} x_{ij}$$

$$\text{(4.5)} \quad \text{s.t.} \quad x(\delta^*(i)) \equiv 0 \bmod 2 \qquad \forall i \in V,$$

$$\text{(4.6)} \quad x(\delta^+(S)) - x(\delta^-(S)) \leq x(\delta(S)) \qquad \forall S \subset V,$$

$$\text{(4.7)} \quad x_e \geq 1 \text{ and integer} \qquad \forall e \in E \cup A,$$

where (4.7) implies that all the links are in the solution, (4.5) ensures that the resulting graph will be even, and (4.6) ensures that the balanced-set conditions will be satisfied. This formulation was proposed by Nobert and Picard [30] and was also used in Corberán, Romero, and Sanchis [11] and Corberán, Mejía, and Sanchis [7].

Note that constraints (4.5) are not linear, and there is no linear expression of them without using integer variables. The effect of constraints (4.5) can be partially achieved by the following odd-cut inequalities (see Corberán, Romero, and Sanchis [11]):

$$x(\delta^*(S)) \geq |\delta^*(S)| + 1 \quad \forall S \subset V \text{ with } |\delta^*(S)| \text{ odd.} \tag{4.8}$$

However, the addition of these inequalities does not guarantee the evenness of the solution as constraints (4.5) do.

Inequalities (4.8) and (4.6), although exponential in number, can be separated in polynomial time. Padberg and Rao [31] proved that violated odd-cut inequalities can be identified in polynomial time. The equivalent result for balanced-set inequalities was proved by Nobert and Picard [30], and a direct proof and a description of the method was presented in Benavent, Corberán, and Sanchis [1].

Now consider a set $S \subset V$ such that $\delta(S) = \emptyset$. Then the balanced-set condition corresponding to S and $V \setminus S$ implies the so-called *symmetry equation* $x(\delta^+(S)) = x(\delta^-(S))$. Hence, the above formulation includes an equation associated with each set $S \subset V$ with $\delta(S) = \emptyset$. Although most of these equations will be linearly dependent, if we consider the q subgraphs of G induced by the edges, it can be shown that any q—1 of the corresponding symmetry equations are linearly independent. In Corberán, Romero, and Sanchis [11] these equations,

$$x(\delta^+(K_i)) = x(\delta^-(K_i)), \quad i = 1, \ldots, q, \tag{4.9}$$

are referred to as *the system equations*, where $K_1, \ldots, K_q$ denote the sets of vertices of the connected components of the graph (V, E).

Let $\text{MCPP}_{F1}(G)$ be the convex hull of all the vectors $x \in \mathbb{R}^{E \cup A}$ satisfying (4.5) to (4.7). Although $\text{MCPP}_{F1}(G)$ has not been studied directly, a generalization of it, the Mixed General Routing Problem (MGRP) polyhedron, has been studied in Corberán, Romero, and Sanchis [11] and Corberán, Mejía, and Sanchis [7] (see also Chapter 6 of this book). Therefore, it is known that $\text{MCPP}_{F1}(G)$ is an unbounded polyhedron and $\dim(\text{MCPP}_{F1}(G)) = |E \cup A| - q + 1$, and that the following inequalities are facet-inducing for $\text{MCPP}_{F1}(G)$:

- trivial inequalities: $x_e \geq 1 \ \forall e \in E \cup A$,
- balanced-set inequalities (4.6),
- odd-cut inequalities (4.8), and
- Odd zigzag inequalities described in Corberán, Plana, and Sanchis [10] (see Section 4.4.2).

4.3.2.2 ▪ Formulation F2

Formulation F2 uses two variables x_{ij} and x_{ji} associated with each edge $e = (i, j)$, representing the number of times edge e is traversed in the corresponding direction. As has been said, this formulation is based on the sufficient conditions for a mixed graph to be Eulerian, and it reads as follows:

$$(4.10)\ (\text{MCPP-F2}) \quad \min \sum_{e=(i,j)\in E} c_e(x_{ij}+x_{ji}) + \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$(4.11)\ \text{s.t.} \quad x_{ij}+x_{ji} \geq 1 \qquad \forall e=(i,j)\in E,$$

$$(4.12) \quad x(\delta^+(i)) - x(\delta^-(i)) + \sum_{(i,j)\in\delta(i)} (x_{ij}-x_{ji}) = 0 \qquad \forall i \in V,$$

$$(4.13) \quad x_{ij} \geq 1 \text{ and integer} \qquad \forall (i,j)\in A,$$

$$(4.14) \quad x_{ij}, x_{ji} \geq 0 \text{ and integer} \qquad \forall e=(i,j)\in E,$$

where (4.11) and (4.13) imply that all the edges and arcs are in the solution, and (4.12) implies that the symmetry conditions will be satisfied. Note that integrality and symmetry conditions imply that the resulting graph will be even. This formulation was proposed by Kappauf and Koehler [24] and is also used in Christofides et al. [5] and Ralphs [36].

Moreover, it is shown in Kappauf and Koehler [24], Ralphs [36], and Win [38] that the components of the extreme points of the polyhedron associated with the linear relaxation of F2 have values of either 0.5 or a nonnegative integer, and that the fractional values occur only on some edges $e=(i,j)$, where $x_{ij}=x_{ji}=0.5$. As with the WPP (Win [38]), this result could be the basis for a simple heuristic algorithm consisting of rounding the variables of value 0.5 to 1. This algorithm could provide good feasible solutions even for large-size instances (mainly on those with a small number of edges) in very short computing times.

Let $\text{MCPP}_{F2}(G)$ be the convex hull of all the vectors $x \in \mathbb{R}^{E\cup A}$ satisfying (4.11) to (4.14). As happens with $\text{MCPP}_{F1}(G)$, $\text{MCPP}_{F2}(G)$ has not been studied directly, but again a generalization of it, the WPP polyhedron in this case, has been (partially) described in Win [38], Grötschel and Win [18], and Corberán et al. [9]. From the results obtained for the WPP, it can be deduced that $\text{MCPP}_{F2}(G)$ is an unbounded polyhedron, with dimension $2|E|+|A|-|V|+1$, and that the following inequalities are facet-inducing:

- trivial inequalities: $x_{ij} \geq 1\ \forall (i,j)\in A$ and $x_{ij}, x_{ji} \geq 0\ \forall e=(i,j)\in E$,
- traversing inequalities (4.11),
- odd-cut inequalities (4.8),
- k-wheel inequalities proposed in Win [38] (see Section 4.4.2), and
- Odd zigzag inequalities presented in Corberán, Plana, and Sanchis [10] (see also Section 4.4.2).

4.3.2.3 ▪ Formulations comparison

It is easy to see that (integer) formulations F1 and F2 are equivalent, but what can be said about their linear programming relaxations? In Corberán, Mota, and Sanchis [8] the lower bounds for the MCPP obtained using LP-based methods are compared. Remember that constraints (4.5) in F1 are not linear, and there is no linear expression of them without using integer variables. Hence, these constraints are removed, and odd-cut inequalities (4.8) are added to F1.

On the other hand, note that once the integrality conditions have been removed from F2, symmetry conditions alone no longer guarantee the evenness of the vertices. Hence, constraints (4.8) are also added to F2.

Therefore, let us call the linear relaxation of F1 LP_{F1},

$$
\begin{aligned}
x_{ij} &\geq 1 && \forall (i,j) \in A, \\
x_e &\geq 1 && \forall e \in E, \\
x(\delta^+(S)) - x(\delta^-(S)) &\leq x(\delta(S)) && \forall S \subset V, \quad (4.6) \\
x(\delta^*(S)) &\geq |\delta^*(S)| + 1 && \forall S \subset V \text{ with } |\delta^*(S)| \text{ odd}, \quad (4.8)
\end{aligned}
$$

and the linear relaxation of F2 LP_{F2},

$$
\begin{aligned}
x_{ij} &\geq 1 && \forall (i,j) \in A, \\
x_{ij}, x_{ji} &\geq 0 && \forall (i,j) \in E, \\
x_{ij} + x_{ji} &\geq 1 && \forall (i,j) \in E, \quad (4.11) \\
x(\delta^*(S)) &\geq |\delta^*(S)| + 1 && \forall S \subset V \text{ with } |\delta^*(S)| \text{ odd}, \quad (4.8) \\
x(\delta^+(i)) - x(\delta^-(i)) + \sum_{(i,j) \in \delta(i)} (x_{ij} - x_{ji}) &= 0 && \forall i \in V. \quad (4.12)
\end{aligned}
$$

Since constraints (4.6) and (4.8) in LP_{F1} and (4.8) in LP_{F2} are exponential in number, only a subset of them can be explicitly added to the LPs to be solved. Consider the following initial LP relaxation, $\mathrm{LP0}_{F1}$, corresponding to LP_{F1} containing

- the system equations (4.9),
- constraints $x_{ij} \geq 1 \ \forall (i,j) \in A$ and $x_e \geq 1 \ \forall e \in E$,
- one balanced-set inequality (4.6) for each "unbalanced" vertex, i.e., a vertex i for which $|\delta(i)| < \big||\delta^+(i)| - |\delta^-(i)|\big|$, and
- one odd-cut inequality (4.8) for each odd "balanced" vertex,

and $\mathrm{LP0}_{F2}$ (corresponding to LP_{F2}) as the LP containing

- constraints $x_{ij} \geq 1 \ \forall (i,j) \in A$ and $x_{ij}, x_{ji} \geq 0 \ \forall (i,j) \in E$,
- constraints (4.11),
- one odd-cut inequality (4.8) for each odd-degree vertex, and
- one symmetry equation (4.12) for each vertex.

In Corberán, Mota, and Sanchis [8] it is shown that $\mathrm{LP0}_{F2}$ is stronger than $\mathrm{LP0}_{F1}$. Regarding the relaxations LP_{F1} and LP_{F2}, note that although they include an exponential number of inequalities, both types of inequalities can be separated in polynomial time. Therefore, the computational effort to solve LP_{F1} and LP_{F2} is, at least in theory, comparable. Moreover, it is shown in [8] that both relaxations produce the same bound.

4.3.3 ▪ Exact algorithms

Several exact procedures have been proposed for the optimal solution of the MCPP. Christofides et al. [5] proposed a branch-and-bound algorithm, using two different lower bounds obtained by relaxing two types of constraints in a Lagrangian manner, able to solve instances with up to 50 vertices, 85 arcs, and 39 edges. Grötschel and Win [19] presented a cutting-plane algorithm for the WPP that was used to solve nine MCPP instances with up to 172 vertices, 116 arcs, and 154 edges. Nobert and Picard [30] were able to solve 148 instances out of 180 instances with a cutting-plane algorithm. The sizes of the instances were in the range $10 \leq |V| \leq 169$, $2 \leq |A| \leq 2896$, and $15 \leq |E| \leq 1849$. Finally, in Corberán et al. [9] a branch-and-cut algorithm for the exact solution of the

WPP is presented. This algorithm, which will be described in Section 4.4.3, was applied to solve MCPP instances and was able to solve 17 out of 24 instances with $|V| = 3000$, $1097 \leq |A| \leq 6742$, and $1992 \leq |E| \leq 6799$ optimally in less than 15 minutes. The average gap obtained on the seven unsolved instances was less than 1%. As far as we know, this is the best exact method for solving the MCPP.

4.4 ▪ The windy postman problem

In this section we define the WPP, present some notation and its formulation, and discuss its heuristic and exact solution.

The WPP can be defined as follows. Given an undirected and connected graph $G = (V,E)$ with two nonnegative costs c_{ij} and c_{ji} associated with each edge $(i,j) \in E$ corresponding to the cost of traversing it from i to j and from j to i, respectively, the WPP is to find a minimum cost tour on G traversing each edge at least once. This problem was introduced by Minieka [28], who wrote *"This* [costs symmetry] *is hardly a good assumption when one direction might be uphill and the other downhill, when one direction might be with the wind and the other against the wind or when fares are different depending on direction."* The WPP contains the MCPP as a special case arising when, for each edge $(i,j) \in E$, either $c_{ij} = c_{ji}$ or $\max\{c_{ij}, c_{ji}\} = \infty$. Therefore, it is easy to see that the WPP is NP-hard (Brucker [2], Guan [21]), although it can be solved in polynomial time if G is even (Win [39]), if the cost of the two opposite orientations of every cycle in G is the same (Guan [21]), or if G is a series-parallel graph (Zaragoza [43]).

If G satisfies that the cost of the two possible orientations of every cycle is the same, the WPP is solved as an undirected CPP where each edge (i,j) has cost $(c_{ij} + c_{ji})/2$. In what follows, we describe the polynomial-time algorithm proposed by Win [39] to find a WPP tour on an even (Eulerian) graph $G = (V,E)$. It is based on the Edmonds and Johnson algorithm for the MCPP on even graphs [13].

Win's algorithm for Eulerian graphs

1 For each edge $(i,j) \in E$, if $c_{ij} \leq c_{ji}$, then orient (i,j) from i to j; otherwise, orient (i,j) from j to i. In this way we obtain a directed graph, $D_G = (V,A)$.

2 Construct a directed graph $D' = (V,A')$ with arc set A', arc costs, arc capacities, and node demands defined as follows:
   - For each arc $(i,j) \in A$ there are three arcs in A', namely, one copy of (i,j) at cost c_{ij} and infinite capacity, one copy of (j,i) at cost c_{ji} and infinite capacity, and one additional copy of (j,i), denoted by $(j,i)'$, at cost $(c_{ij} - c_{ji})/2$ and capacity two (such arcs are called "artificial" arcs).
   - For each node $i \in V$, the demand is $d_i = d^+(i) - d^-(i)$ in D_G (negative demands are interpreted as supplies).

3 Find a minimum-cost flow in D' satisfying the demands and supplies of the vertices.

4 Denote by y^*_{ij}, y^*_{ji}, and $y^*_{(ji)'}$ the flow values along the arcs (i,j), (j,i), and $(j,i)'$, respectively. For every three arcs $(i,j),(j,i),(j,i)' \in A'$ proceed as follows: If $y^*_{(ji)'} = 0$, then put $y^*_{ij} + 1$ copies of (i,j) (and no copy of (j,i)) in A''; otherwise (i.e., if $y^*_{(ji)'} = 2$), put $y^*_{ji} + 1$ copies of (j,i) (and

no copy of (i,j)) in A''. $D''=(V,A'')$ is Eulerian, and any Eulerian directed walk in D'' is an optimal WPP tour of G.

4.4.1 ▪ Heuristic algorithms

As in the MCPP, the above algorithm for even graphs is the basis for a heuristic procedure for the general case (Win [39]). It consists of transforming first the original graph into an even one, and then applying the previous algorithm for Eulerian graphs.

Win's algorithm for general graphs

1 Transform G into an Eulerian graph.
   - For each edge $e=(i,j)\in E$, define $\bar{c}_e=(c_{ij}+c_{ji})/2$.
   - Identify odd nodes of G, and compute the shortest path distances between every pair of them using distances $\bar{c}_e$.
   - Find a minimum-cost perfect matching of the odd nodes.
   - Add to E one additional copy of each edge that lies on the shortest path joining two matched odd nodes. The resulting graph, say, $G'=(V,E')$, is Eulerian.

2 Apply Win's algorithm for Eulerian graphs to G' with the original cost function c_e.

Win [39] proved that this algorithm has a worst-case bound with a value of twice the optimum, which is approachable. Win [38] proposed another heuristic entailing rounding up the optimal solutions to the linear relaxation of the WPP formulation presented below. This algorithm was based on the following result: The values of the variables in any optimal LP solution are integer or 0.5, and $x_{ij}=0.5$ if and only if $x_{ji}=0.5$ (Kappauf and Koehler [24], Win [38]). Then, it can be proved that forcing $x_{ij}=x_{ji}=1$ for these variables leads to a feasible WPP solution. Win [38] also proved that this heuristic again has a worst-case ratio of 2 and the bound is approachable. Based on their algorithm for the MCPP, Raghavachari and Veerasamy [35] modify Win's LP algorithm to get a new one with a better worst-case ratio of 3/2.

4.4.2 ▪ Problem formulation

Let $G=(V,E)$ be the graph for which a minimum-cost WPP tour has to be determined. For $S_1,S_2\subseteq V$, $(S_1:S_2)$ denotes the set of edges with one endpoint in S_1 and the other in S_2. Given $S\subseteq V$, $\delta(S)=(S:V\setminus S)$ and $E(S)=(S:S)$.

Let x_{ij} be the number of times edge (i,j) is traversed from i to j in a WPP tour. For $F\subseteq E$, we define $x(F)=\sum_{(i,j)\in F}(x_{ij}+x_{ji})$, and for $S_1,S_2\subset V$, we define

$$x(S_1,S_2)=\sum_{i\in S_1,j\in S_2}x_{ij}.$$

Note that $x(S_1:S_2)=x(S_1,S_2)+x(S_2,S_1)$.

The ILP formulation of the WPP in Win [38] and Grötschel and Win [19] is

$$\text{(4.15)} \qquad \text{(WPP)} \quad \min \sum_{(i,j)\in E} (c_{ij}x_{ij} + c_{ji}x_{ji})$$

$$\text{(4.16)} \qquad \text{s.t.} \quad x_{ij} + x_{ji} \geq 1 \qquad \forall (i,j) \in E,$$

$$\text{(4.17)} \qquad \sum_{(i,j)\in\delta(i)} (x_{ij} - x_{ji}) = 0 \qquad \forall i \in V,$$

$$\text{(4.18)} \qquad x_{ij}, x_{ji} \geq 0 \text{ and integer} \qquad \forall (i,j) \in E,$$

where conditions (4.16) imply that each edge will be traversed at least once and conditions (4.17) force the (directed) graph represented by the tour to be symmetric. The above system includes an equation associated with each vertex. The $|V|$ equations (4.17) will be referred to as the *symmetry equations*, and any $|V|-1$ of them are linearly independent.

Let $\text{WPP}(G) \subseteq \mathbb{Z}^{2|E|}$ be the convex hull of vectors satisfying (4.16) to (4.18). In Win [38] and Grötschel and Win [18], it is shown that $\text{WPP}(G)$ is an unbounded polyhedron, with dimension $2|E|-|V|+1$, and that trivial and traversing inequalities are facet-inducing.

In addition to these inequalities, other families of inequalities that are not present in the formulation are known.

Odd-cut inequalities

Odd-cut inequalities were presented in Win [38] and shown to be facet-inducing for $\text{WPP}(G)$. As has been said, they are based on the fact that any edge cutset must be traversed an even number of times:

$$\text{(4.19)} \qquad x(\delta(S)) \geq |\delta(S)| + 1 \quad \forall\, S \subset V,\ |\delta(S)| \text{ odd}.$$

k-wheel inequalities

A wheel consists of a cycle (called the rim), every node of which is incident with a common node (called the center). An edge joining the center and a node on the rim is called a spoke. A k-wheel, denoted by W_k, is a wheel whose rim contains k nodes. Figure 4.1(a) shows a 3-wheel W_3 in which each edge has been oriented in such a way that the edges of the rim do not form a directed cycle and all the spokes are oriented to the center. Then, the 3-wheel inequality

$$\text{(4.20)} \qquad x_{12} + x_{13} + x_{14} + x_{32} + x_{42} + x_{43} \geq 3$$

is valid and facet-inducing for the corresponding WPP polyhedron (Win [38]).

This inequality can be generalized by replacing each vertex by a connected subgraph and each arc by a subset of parallel arcs with the same direction in the following way (see Figure 4.1b): Consider a partition of the set of vertices V into four subsets, M^1, M^2, M^3, and M^4, where each M^i contains an odd number of odd vertices and each subgraph $G(M^i)$ is connected, and let $x(A')$ be the set of variables corresponding to the traversal of the edges in the direction depicted in Figure 4.1(b). The 3-wheel inequality is then

$$\text{(4.21)} \qquad x(A') \geq \frac{1}{2}\Big(|(M^1 : M^3)| + |(M^2 : M^4)| + |(M^1 : M^4)| + |(M^2 : M^3)|\Big) + 1.$$

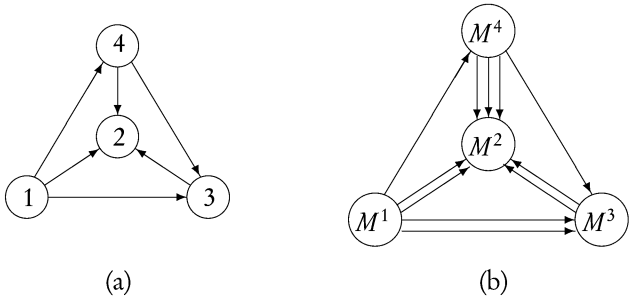


Figure 4.1. *3-wheel inequalities.*

The 3-wheel inequalities can be extended to the more general k-wheel inequalities, which contain an odd number k of nodes in the rim (see details in Win [38]).

Odd zigzag inequalities

When solving the WPP, it is not unusual to find fractional solutions containing closed walks, resembling a zigzag, with $x_{ij} + x_{ji} = 1.5$, such as the vector represented in Figure 4.2(a). This kind of solution violates another family of facet-inducing inequalities for the WPP, the Odd zigzag inequalities, presented in Corberán, Plana, and Sanchis [10]. These inequalities generalize the 3-wheel inequalities.

As with the k-wheel inequalities, consider a partition of the set of vertices V into four subsets, M^1, M^2, M^3, and M^4, where each M^i contains an odd number of odd vertices. Let us define $\mathscr{H} = (M^1 : M^2) \cup (M^3 : M^4)$ (horizontal edges) and $\mathscr{D} = (M^2 : M^3) \cup (M^1 : M^4)$ (diagonal edges). Note that $\mathscr{H} \cup \mathscr{D} = \delta(M^1 \cup M^3)$. Let us assume we have a subset of edges $\mathscr{F} \subset (\mathscr{H} \cup \mathscr{D})$ satisfying $|\mathscr{H} \setminus \mathscr{F}| + |\mathscr{D} \cap \mathscr{F}| = |\mathscr{D} \setminus \mathscr{F}| + |\mathscr{H} \cap \mathscr{F}|$, or, equivalently,

$$|\mathscr{H}| + |\mathscr{D}| = 2|\mathscr{H} \cap \mathscr{F}| + 2|\mathscr{D} \setminus \mathscr{F}|$$

(see Figure 4.2(b), where edges in $\mathscr{F}$ are represented in bold lines).

In Corberán, Plana, and Sanchis [10], it is shown that the following Odd zigzag inequality, whose coefficients are shown in Figure 4.2(b), is valid and facet-inducing:

$$\begin{aligned} (4.22) \qquad & x(\delta(M^1 \cup M^2)) + 2x(M^2, M^1) + 2x(M^4, M^3) + 2x(F_{zz}) \\ & \geq |(M^1 : M^3)| + |(M^2 : M^4)| + |(M^1 : M^4)| + |(M^2 : M^3)| + 2|\mathscr{H} \cap \mathscr{F}| + 2, \end{aligned}$$

where $x(F_{zz})$ denotes the variables associated with the edges in $\mathscr{F}$ in the direction given by the zigzag, i.e., in the direction $M^1 \to M^2 \to M^3 \to M^4 \to M^1$.

Set $\mathscr{F}$ can be understood in the following way. A fractional solution, such as the one in Figure 4.2(a), defines an orientation of the edges in $\delta(M^1 \cup M^3)$. Given any such orientation, set $\mathscr{F}$ is defined by all the edges that have been oriented in the opposite direction to the zigzag, i.e., in the direction $M^4 \to M^3 \to M^2 \to M^1 \to M^4$. In particular, set $\mathscr{F}$ in Figure 4.2(b) is defined from the orientation associated with the fractional solution shown in Figure 4.2(a). Other sets $\mathscr{F}$ can be defined to obtain valid inequalities, but only the one shown in Figure 4.2(b) has an associated inequality violated by the fractional solution.

Note that in the special case when $\mathscr{F} = \emptyset$, the Odd zigzag inequalities are equivalent to the 3-wheel inequalities, and hence they induce the same facets. However, note that

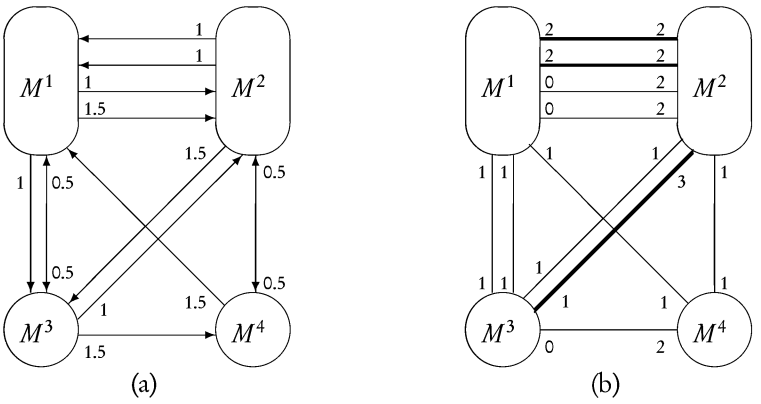


Figure 4.2. *Fractional solution and Odd zigzag inequality.*

when $\mathcal{F} \neq \emptyset$, Odd zigzag inequalities are a different class of facet-inducing inequalities for the WPP. In fact, as noted in Corberán, Plana, and Sanchis [10], it can be seen that the fractional solution shown in Win's Thesis (see Figure 4.3), which satisfies all the 3-wheel inequalities, violates the Odd zigzag inequality defined by node sets $M^1 = \{6\}$, $M^2 = \{3,4,5\}$, $M_3 = \{2\}$, $M_4 = \{1\}$, and edge set $F = \{e_{46}\}$, which is depicted in Figure 4.3. It can be seen that the Odd zigzag inequality is $F(x) \geq 8$, while the fractional solution x^* satisfies $F(x^*) = 7$.

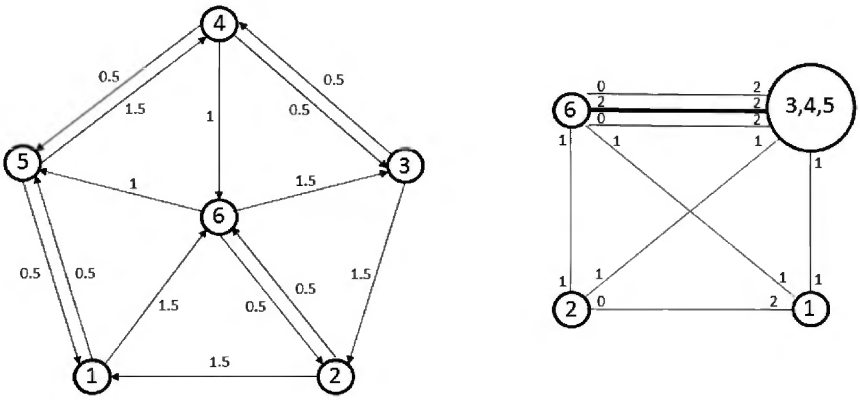


Figure 4.3. *Fractional solution from Win's Thesis and the Odd zigzag inequality.*

It is worth noting that, unlike other Arc Routing Polyhedra, WPP(G) has facet-inducing inequalities, such as the Odd zigzag inequalities presented above, that are not strictly configuration inequalities (Naddef and Rinaldi [29]) since they can have variables x_{uv}, x_{st}, with $u, s \in B_i$ and $v, t \in B_j$, with different coefficients.

Even-even and Odd-odd zigzag inequalities

In Corberán et al. [9] the authors describe some fractional solutions having four edges with value 1.5 forming a nondirected path joining two odd nodes with two even nodes

(such as those in Figures 4.4(a) and 4.5(a), that cannot be separated by an Odd zigzag inequality. Two new versions of zigzag inequalities cutting these fractional solutions are presented in that paper. The name of the inequalities refers to the degree of the two shores of the edge cutset $\delta(M^1 \cup M^3)$ (see Figures 4.4(b) and 4.5(b), where a set M^i represented with a double circle is odd).

For the Even-even zigzag inequalities, sets M^2 and M^4 are odd, sets M^1 and M^3 are even, and $\mathcal{F}$ is a subset of edges, $\mathcal{F} \subset \delta(M^1 \cup M^3)$ (shown as bold lines in Figure 4.4a), satisfying

$$|(M^1 : M^2 \cup M^4) \setminus \mathcal{F}| + |(M^2 \cup M^4 : M^3) \cap \mathcal{F}| = |(M^1 : M^2 \cup M^4) \cap \mathcal{F}| + |(M^2 \cup M^4 : M^3) \setminus \mathcal{F}|. \tag{4.23}$$

The corresponding Even-even zigzag inequality is

$$x(\delta(M^3)) + 2x(M^2 : M^4) + 2x(M^2, M^1) + 2x(M^4, M^1) + 2x(F_{zz}) \geq |(M^1 : M^3)| + 2|(M^2 : M^4)| + |(M^2 : M^3)| + |(M^3 : M^4)| + 2|\delta(M^1) \cap \mathcal{F}| + 2, \tag{4.24}$$

where $x(F_{zz})$ denotes the variables associated with the edges in $\mathcal{F}$ in the direction of the zigzag, i.e., $M^1 \to M^2 \to M^3$ and $M^1 \to M^4 \to M^3$. The coefficients of the variables are also shown in Figure 4.4(b).

For the Odd-odd zigzag inequalities (see Figure 4.5(b)), sets M^2 and M^3 are odd, sets M^1 and M^4 are even, set $\mathcal{F} \subset \delta(M^1 \cup M^3)$ satisfies the condition

$$|(M^1 : M^2 \cup M^4) \setminus \mathcal{F}| + |(M^3 : M^4) \setminus \mathcal{F}| + |(M^2 : M^3) \cap \mathcal{F}| + 1 = |(M^1 : M^2 \cup M^4) \cap \mathcal{F}| + |(M^3 : M^4) \cap \mathcal{F}| + |(M^2 : M^3) \setminus \mathcal{F}|, \tag{4.25}$$

and the corresponding Odd-odd zigzag inequality is

$$x(M^2 \cup M^4, M^1 \cup M^3) + x(M^1 : M^3) + x(M^2 : M^4) + x(M^3, M^2) + x(F_{zz}) \geq |(M^1 : M^3)| + |(M^2 : M^4)| + |(M^2 : M^3) \setminus \mathcal{F}| + |\mathcal{F}| + 1, \tag{4.26}$$

where, again, $x(F_{zz})$ denotes the variables associated with the edges in $\mathcal{F}$ in the direction of the zigzag, i.e., $M^1 \to M^2 \to M^3 \to M^4$ and $M^1 \to M^4$.

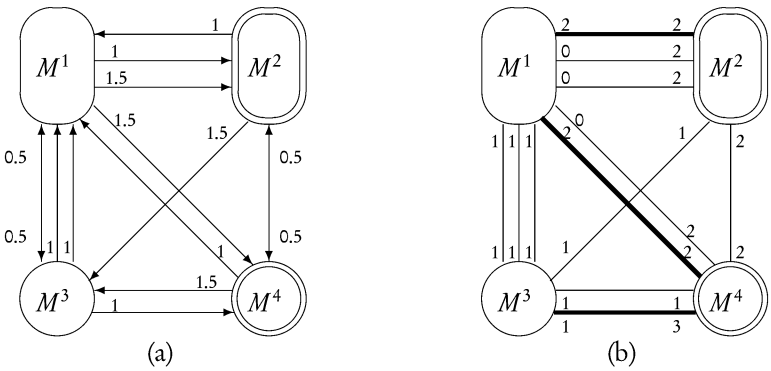


Figure 4.4. *A fractional solution and an Even-even zigzag inequality.*

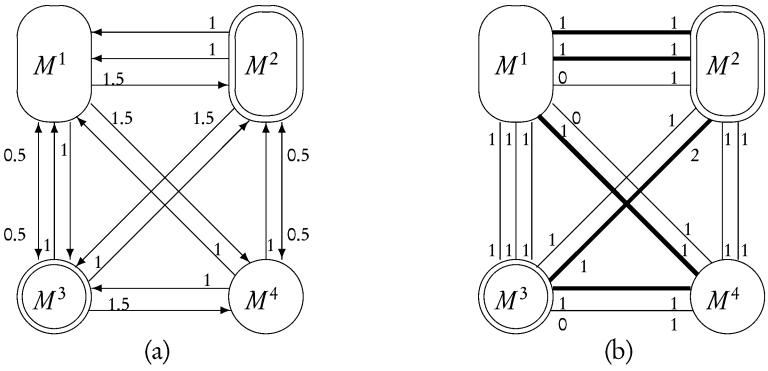


Figure 4.5. *A fractional solution and an Odd-odd zigzag inequality.*

4.4.3 ▪ Exact algorithms

In Corberán et al. [9] a branch-and-cut algorithm for the WPP is presented. This procedure uses heuristic and exact procedures for separating violated odd-cut inequalities and incorporates heuristic separation algorithms for all the families of zigzag inequalities. Moreover, since in the same paper it is shown that all zigzag inequalities are Chvátal–Gomory inequalities of rank at most 2, a polynomial time algorithm for identifying maximally violated mod-k inequalities is added. Note that a mod-k cut (see Caprara, Fischetti, and Letchford [3]) is a rank-1 Chvátal–Gomory inequality in which the multipliers are restricted to the set $\{0, \frac{1}{k}, \ldots, \frac{k-1}{k}\}$. Basically, the method consists of solving a system of congruences where the system is composed of all the binding inequalities of the last LP plus an inequality for each symmetry equation. Moreover, in order to get good upper bounds that decrease the size of the search tree, a heuristic algorithm based on the information given by the fractional solution x^* obtained after processing each node has been implemented.

At the time of writing this chapter, this is the best exact algorithm that has been published for the WPP, and it is capable of solving very large WPP instances with up to 3000 nodes and 9000 edges. Of the 120 instances it was tested on, the algorithm solved 91 out of 96 instances with $500 \leq |V| \leq 2000$ and $813 \leq |E| \leq 6129$ in less than two hours on average, and eight out of 24 instances with $|V| = 3000$ and $4986 \leq |E| \leq 9085$. For the unsolved instances, the average gap obtained was less than 0.2%. Tests were run on an Intel Pentium IV 2.80GHz and 2GB RAM with a time limit of 10 hours. When compared with the results obtained by the same algorithm in MCPP instances of a similar size, the results obtained are slightly worse, which seems to imply that the WPP is harder than the MCPP.

A completely different approach to the WPP solution was proposed by Yan and Thompson [40]. The authors slightly modify the WPP formulation to get an equivalent set covering problem. Taking into account this transformation, an exact algorithm and a heuristic algorithm producing good results for instances with up to 90 nodes and 180 edges are proposed. Both methods are based on a branch-and-bound procedure previously designed for the solution of set covering, partitioning, and packing problems.

4.5 ▪ Related problems

The MCPP can be generalized by providing, for each arc and edge e, two integers u_e, $l_e \geq 0$, and requiring that e is used at least l_e and at most u_e times. This problem, called the *Bounded Mixed Postman Problem*, was suggested by Edmonds and Johnson [13] and studied by Zaragoza [42]. Another related problem mentioned in [42] is the *Restricted Mixed Postman Problem*, where given a subset $R \subseteq E \cup A$ of restricted edges and arcs, the problem consists of finding a restricted tour, i.e., a tour using each restricted edge and arc exactly once. The special cases when $R = A$ and $R = E$ are called the *Edges Postman Problem* and the *Arcs Postman Problem*, respectively, and have been studied by Veerasamy [37] and Zaragoza [42].

Given a mixed graph G, a cycle cover of G is a family $C = \{C_1, \ldots, C_k\}$ of cycles of G such that each edge and arc of G belongs to at least one cycle in C. The weight of C is $\sum_{i=1}^{k} |C_i|$. The *Minimum Cycle Cover Problem* (MCC) is as follows: Given a strongly connected mixed graph G without bridges, find a cycle cover of G with weight as small as possible. In Lee and Wakabayashi [25] it is proved that this problem is NP-hard if G is a planar graph. However, the MCC (and the MCPP) defined on mixed graphs with bounded treewidth can be solved in polynomial time (Fernandes, Lee, and Wakabayashi [15]).

A problem related to the WPP on Eulerian graphs is the *Minimum Cost Eulerian Orientation Problem* (MCEOP). Let $G = (V, E)$ be an undirected Eulerian graph with two costs c_{ij} and c_{ji} associated with each edge $e = (i, j) \in E$, where c_{ij} and c_{ji} are the costs of orienting edge e as (i, j) or (j, i), respectively. An orientation of G is obtained by replacing edge e with arc (i, j) or arc (j, i). The MCEOP consists of finding the minimum cost Eulerian orientation of G. Note that the MCEOP is a WPP with an additional restriction: that we should traverse each edge e only once (either from i to j or from j to i). Guan and Pulleyblank [22] proposed several polynomial-time algorithms for this problem. Another polynomial-time algorithm for the MCEOP, similar to Win's algorithm for Eulerian graphs described in Section 4.4.1, as well as a complete description of its associated polytope, can be found in Win [38, 39].

Another problem related to the WPP is the *Plowing with Precedence Problem* (PPP) proposed by Dussault et al. [12]. This is a variant of the WPP motivated by the fact that deadhead travel over streets that have already been plowed is significantly faster than the time it takes to plow the street. The authors incorporate the observation that on some steep streets it is much more difficult, or even impossible, to plow uphill. The idea is to design routes that try to avoid plowing uphill on steep streets and take advantage of the faster traversal time on plowed streets. Note that this assumption requires the possibility of plowing against the direction of traffic and allows for the possibility of deadhead travel when only one side of the street has been plowed, regardless of the direction. For example, if a steep street has been plowed downhill, it is possible to traverse the street uphill without plowing as long as the vehicle eventually returns to clear the other side. If a street has not been plowed, however, then it is impossible to deadhead that street because the snow is too deep to traverse without plowing. Let $G = (V, E)$ be an undirected graph. Each edge $e = (i, j)$ has four costs: c_{ij}^{+}, the cost of plowing e from i to j; c_{ji}^{+}, the cost of plowing e from j to i; c_{ij}^{-}, the cost of traversing e from i to j after plowing; and c_{ji}^{-}, the cost of traversing e from j to i after plowing. If vertex j has a higher altitude than vertex i, then usually $c_{ij}^{+} >> c_{ji}^{+} >> c_{ij}^{-} \geq c_{ji}^{-}$. The PPP consists of finding the route that begins and ends at the depot (vertex 0), plows each edge twice (for each side of the street), and minimizes total cost. The order of a route is important since the cost of traversing an edge depends on

whether or not the edge has been traversed previously; i.e., it is not possible to deadhead an edge $e = (i, j)$ from i to j, for instance, at cost c_{ij}^- if we have not previously plowed e at cost c_{ij}^+ or c_{ji}^+. Therefore, precedence is important in the PPP. Dussault et al. [12] formulate the PPP as an integer program and propose a local search procedure producing very good solutions. The algorithm optimally solves a relaxation of the PPP in which precedence is ignored. This relaxed problem is similar to the WPP. The optimal solution to this problem, called a partial solution by the authors, gives the number of times to plow and deadhead each edge in each direction and is used to construct a route for the PPP. Starting with this partial solution, an initial route using an alternative to Fleury's algorithm is constructed. This initial route is optimal if precedence is satisfied; otherwise, a local search heuristic is applied that modifies the route while taking precedence into account.

Acknowledgments

The authors wish to thank the Spanish Ministerio of Economía y Competitividad (project MTM2012-36163-C06-02) and the Generalitat Valenciana (project GVPROMETEO2013-049) for their support.

Bibliography

[1] E. Benavent, Á. Corberán, and J.M. Sanchis, *Linear programming based methods for solving arc routing problems*, in Arc Routing: Theory, Solutions and Applications, M. Dror, ed., Kluwer, Boston, 2000, pp. 231–275.

[2] P. Brucker, *The Chinese postman problem for mixed graphs*, in Proceedings of the International Workshop, Bad Honnef, 1980, Lecture Notes in Computer Science 100, Springer, Berlin, 1981, pp. 354–366.

[3] A. Caprara, M. Fischetti, and A.N. Letchford, *On the separation of maximally violated mod-k cuts*, Mathematical Programming, 87 (2000), pp. 37–56.

[4] N. Christofides, *The optimum traversal of a graph*, Omega, 1 (1973), pp. 719–732.

[5] N. Christofides, E. Benavent, V. Campos, Á. Corberán, and E. Mota, *An optimal method for the mixed postman problem*, in System Modelling and Optimization, P. Thoft-Christensen, ed., Lecture Notes in Control and Information Sciences 59, Springer, Berlin, 1984, pp. 641–649.

[6] Á. Corberán, R. Martí, and J.M. Sanchis, *A GRASP heuristic for the mixed Chinese postman problem*, European Journal of Operational Research, 142 (2002), pp. 70–80.

[7] Á. Corberán, G. Mejía, and J.M. Sanchis, *New results on the mixed general routing problem*, Operations Research, 53 (2005), pp. 363–376.

[8] Á. Corberán, E. Mota, and J.M. Sanchis, *A comparison of two different formulations for arc routing problems on mixed graphs*, Computers & Operations Research, 33 (2006), pp. 3384–3402.

[9] Á. Corberán, M. Oswald, I. Plana, G. Reinelt, and J.M. Sanchis, *New results on the windy postman problem*, Mathematical Programming, 132 (2012), pp. 309–332.

[10] Á. CORBERÁN, I. PLANA, AND J.M. SANCHIS, *Zigzag inequalities: A new class of facet-inducing inequalities for arc routing problems*, Mathematical Programming, 108 (2006), pp. 79–96.

[11] Á. CORBERÁN, A. ROMERO, AND J.M. SANCHIS, *The mixed general routing polyhedron*, Mathematical Programming, 96 (2003), pp. 103–137.

[12] B. DUSSAULT, B. GOLDEN, C. GRÖER, AND E. WASIL, *Plowing with precedence: A variant of the windy postman problem*, Computers & Operations Research, 40 (2013), pp. 1047–1059.

[13] J. EDMONDS AND E.L. JOHNSON, *Matching, Euler tours and the Chinese postman problem*, Mathematical Programming, 5 (1973), pp. 88–124.

[14] H.A. EISELT, M. GENDREAU, AND G. LAPORTE, *Arc routing problems, part 1: The Chinese postman problem*, Operations Research, 43 (1995), pp. 231–242.

[15] C.G. FERNANDES, O. LEE, AND Y. WAKABAYASHI, *Minimum cycle cover and Chinese postman problems on mixed graphs with bounded tree-width*, Discrete Applied Mathematics, 157 (2009), pp. 272–279.

[16] L.R. FORD AND D.R. FULKERSON, *Flows in Networks*, Princeton University Press, Princeton, NJ, 1962.

[17] G.N. FREDERICKSON, *Approximation algorithms for some postman problems*, Journal of the Association for Computing Machinery, 26 (1979), pp. 538–554.

[18] M. GRÖTSCHEL AND Z. WIN, *On the windy postman polyhedron*, Technical report 75, University of Augsburg, Augsburg, Germany, 1988.

[19] ——, *A cutting plane algorithm for the windy postman problem*, Mathematical Programming, 55 (1992), pp. 339–358.

[20] M.G. GUAN, *Graphic programming using odd or even points*, Chinese Mathematics, 1 (1962), pp. 273–277.

[21] ——, *On the windy postman problem*, Discrete Applied Mathematics, 9 (1984), pp. 41–46.

[22] M. GUAN AND W. PULLEYBLANK, *Eulerian orientations and circulations*, SIAM Journal on Algebraic Discrete Methods, 6 (1985), pp. 657–664.

[23] A. HERTZ AND M. MITTAZ, *Heuristic algorithms*, in Arc Routing: Theory, Solutions and Applications, M. Dror, ed., Kluwer, Boston, 2000, pp. 327–386.

[24] C.H. KAPPAUF AND G.J. KOEHLER, *The mixed postman problem*, Discrete Applied Mathematics, 1 (1979), pp. 89–103.

[25] O. LEE AND Y. WAKABAYASHI, *On the circuit cover problem for mixed graphs*, Combinatorics, Probability and Computing, 11 (2002), pp. 43–59.

[26] T.M. LIEBLING, *Graphentheorie in Planungs- und Tourenproblemen*, Lecture Notes in Operations Research and Mathematical Systems 21, Springer, Berlin, 1970.

[27] E. MINIEKA, *Optimization Algorithms for Networks and Graphs*, Marcel Dekker, New York, 1979.

[28] ——, *The Chinese postman problem for mixed networks*, Management Science, 25 (1979), pp. 643–648.

[29] D. NADDEF AND G. RINALDI, *The symmetric traveling salesman polytope and its graphical relaxation: Composition of valid inequalities*, Mathematical Programming, 51 (1991), pp. 359–400.

[30] Y. NOBERT AND J.-C. PICARD, *An optimal algorithm for the mixed Chinese postman problem*, Networks, 27 (1996), pp. 95–108.

[31] M. W. PADBERG AND M. R. RAO, *Odd minimum cut-sets and b-matchings*, Mathematics of Operations Research, 7 (1982), pp. 67–80.

[32] C.H. PAPADIMITRIOU, *On the complexity of edge traversing*, Journal of the Association for Computing Machinery, 23 (1976), pp. 544–554.

[33] W.L. PEARN AND J.B. CHOU, *Improved solutions for the Chinese postman problem on mixed networks*, Computers & Operations Research, 26 (1999), pp. 819–827.

[34] B. RAGHAVACHARI AND J. VEERASAMY, *A 3/2-approximation algorithm for the mixed postman problem*, SIAM Journal on Discrete Mathematics, 12 (1999), pp. 425–433.

[35] ——, *A 3/2-approximation algorithm for the windy postman problem*, Technical report, University of Texas, Dallas, TX, 2001.

[36] T.K. RALPHS, *On the mixed Chinese postman problem*, Operations Research Letters, 14 (1993), pp. 123–127.

[37] J. VEERASAMY, *Approximation Algorithms for Postman Problems*, Ph.D. thesis, University of Texas, Dallas, TX, 1999.

[38] Z. WIN, *Contributions to Routing Problems*, Ph.D. thesis, University of Augsburg, Augsburg, Germany, 1987.

[39] ——, *On the windy postman problem on Eulerian graphs*, Mathematical Programming, 44 (1989), pp. 97–112.

[40] H. YAN AND G. THOMPSON, *Finding postal carrier walk paths in mixed graphs*, Computational Optimization and Applications, 9 (1998), pp. 229–247.

[41] K. YAOYUENYONG, P. CHARNSETHIKUL, AND V. CHANKONG, *A heuristic algorithm for the mixed Chinese postman problem*, Optimization and Engineering, 3 (2002), pp. 157–187.

[42] F. J. ZARAGOZA, *Postman Problems on Mixed Graphs*, Ph.D. thesis, University of Waterloo, Waterloo, Ontario, Canada, 2003.

[43] ——, *Series-parallel graphs are windy postman perfect*, Discrete Mathematics, 308 (2008), pp. 1366–1374.

Chapter 5

The Undirected Rural Postman Problem

Gianpaolo Ghiani
Gilbert Laporte

5.1 ▪ Introduction

The undirected *Rural Postman Problem* (RPP) is defined on an undirected graph $G = (V, E)$, where V is the vertex set and E is the edge set. A subset E_R of the edges are *required*, which means they must be part of the solution. With each edge $e = (i, j)$ is associated a nonnegative cost c_{ij}. The RPP consists of designing a least cost tour traversing each edge of E_R at least once. Equivalently, the problem consists of determining a least cost set of deadheaded edges which, together with the edges of E_R, will yield a Eulerian graph.

The RPP was introduced by Orloff [32] and was proved to be NP-hard by Lenstra and Rinnooy Kan [27]. However, when the graph induced by the edges of E_R is connected, the RPP reduces to an undirected *Chinese Postman Problem* (CPP) (see Chapter 3) and can therefore be solved in polynomial time. Applications of the RPP arise in contexts where the edges of a graph must be serviced by an uncapacitated vehicle. There exist applications of the problem related to the control of plotting or drilling machines [23] and to the optimization of laser-plotter beam movements [19]. In addition, in the solution of a *Capacitated Arc Routing Problem* (see Chapter 7), every route can be optimized individually by means of an RPP algorithm.

A problem closely related to the RPP is the *General Routing Problem* (GRP) defined in Chapter 6. Since the GRP is a generalization of the RPP, many results obtained for the GRP can be applied to the RPP.

The remainder of this chapter is organized as follows. In Section 5.2 we recall some properties of RPP solutions. Mathematical formulations are presented in Section 5.3, followed by exact algorithms in Section 5.4. Constructive and improvement heuristics are described in Section 5.5. Finally, some variants and extensions of the RPP are introduced in Section 5.6.

5.2 ▪ Properties

An RPP optimal solution satisfies a number of properties. Let C_i $(i = 1,\ldots,p)$ be the ith connected component of the subgraph induced by E_R and V_i $(i = 1,\ldots,p)$ its set of vertices. Moreover, let $V_R = \cup_{i=1}^{p} V_i$.

Christofides et al. [5] and a number of subsequent authors have worked on a transformed graph in which only the vertices of V_R appear: First, they add to $G_R = (V_R, E_R)$ an edge between every pair of vertices of V_R having a cost equal to the shortest path length on G; they then simplify the graph by deleting one of the two parallel edges if they have the same cost, as well as all edges $e = (i,j) \notin E_R$ such that $c_{ij} = c_{ik} + c_{kj}$ for some $k \in V$. In the remainder of the chapter, we will work also on the transformed graph satisfying $V_R = V$.

Dominance Relation 1. *In every optimal solution, the maximum number of additional copies of edge e that must be added to G to make it Eulerian is equal to 1 if $e \in E_R$, or to 2 otherwise [5].*

Dominance Relation 2. *Let $e = (i,j)$ be an edge such that i and j belong to some connected component C_i. Then, in every optimal solution, the number of additional copies of e is equal to 1 [9].*

For the edges $e \in E_R$, these two dominance relations are equivalent.

Dominance Relation 3. *Let $e^{(1)}, e^{(2)}, \ldots, e^{(l)}$ be the edges having exactly one vertex in a given component C_i and one vertex in another given component C_j $(i,j \in \{1,\ldots,p\}, i \neq j)$. In any optimal solution, only an edge $e^{(r)}$ such that $c(e^{(r)}) = min\{c(e^{(1)}), c(e^{(2)}), \ldots, c(e^{(l)})\}$ can be deadheaded twice [21].*

Consequently, no more than $p(p-1)/2$ edges need to appear twice in an optimal solution. Tighter properties can be derived as follows.

Dominance Relation 4. *There exists an optimal solution to the RPP in which at most $p-1$ edges are deadheaded twice [21].*

Dominance Relation 5. *Let $G_C = (V_C, E_C)$ be an auxiliary graph having a vertex i' for every component C_i and, for any pair of components C_i and C_j, an edge $e = (i', j')$ corresponding to a least cost edge between C_i and C_j. The only edges that can be deadheaded twice are those belonging to a minimum spanning tree on G_C [21].*

5.3 ▪ Mathematical formulations

The first formulation of the RPP is due to Christofides et al. [5]. Given a set of vertices $S \subset V$, let $\delta(S)$ be the set of edges of E with one extremity in S and one extremity in $V \backslash S$. If $S = \{i\}$, we simply write $\delta(i)$ instead of $\delta(\{i\})$. In the formulation proposed by Christofides et al., variable x_e represents the number of additional copies of edge e that must be added to G to make it Eulerian, and $2z_i$ represents the degree of vertex i:

$$\text{(RPP}^{\text{(CHR)}}\text{)} \qquad \text{minimize} \sum_{e \in E_R} c_e(1 + x_e) + \sum_{e \in E \backslash E_R} c_e x_e \tag{5.1}$$

$$\text{(5.2)} \qquad \text{s.t.} \quad \sum_{e\in\delta(i)\cap E_R}(1+x_e)+\sum_{e\in\delta(i)\cap E\setminus E_R}x_e=2z_i, \qquad i\in V,$$

$$\text{(5.3)} \qquad \sum_{e\in\delta(S)}x_e\geq 2, \qquad S=\cup_{k\in P}V_k, \quad P\subset\{1,\ldots,p\}, P\neq\emptyset,$$

$$\text{(5.4)} \qquad x_e\geq 0 \text{ and integer}, \qquad e\in E,$$

$$\text{(5.5)} \qquad z_i\geq 0 \text{ and integer}, \qquad i\in V.$$

In this formulation, constraints (5.2), (5.4), and (5.5) state that each vertex of V_R must have an even degree, while constraints (5.3) force the solution graph to be connected. The resulting graph will therefore be Eulerian.

In the formulation of Corberán and Sanchis [9], a vertex set $S\in V$ is said to be R-odd (R-even) if an odd (even) number of required edges are incident to S and, again, x_e represents the number of deadheadings of edge e:

$$\text{(5.6)} \qquad (\text{RPP}^{(\text{CORB})}) \qquad \text{minimize}\sum_{e\in E}c_e x_e$$

$$\text{(5.7)} \qquad \text{s.t.} \quad \sum_{e\in\delta(i)}x_e=0 \bmod 2 \qquad \text{if } i\in V \text{ is } R\text{-even},$$

$$\text{(5.8)} \qquad \sum_{e\in\delta(i)}x_e=1 \bmod 2 \qquad \text{if } i\in V \text{ is } R\text{-odd},$$

$$\text{(5.9)} \qquad \sum_{e\in\delta(S)}x_e\geq 2, \qquad S=\cup_{k\in P}V_k, P\subset\{1,\ldots,p\}, P\neq\emptyset,$$

$$\text{(5.10)} \qquad x_e\geq 0 \text{ and integer}, \qquad e\in E.$$

As in the previous formulation, the constraints force the degree of each vertex to be even and the graph to be connected. It is worth noting that this formulation is nonlinear because of the modulo operations in contraints (5.7) and (5.8).

Ghiani and Laporte [21] take advantage of Dominance Relation 5 to simplify this formulation. Let E_{012} (E_{01}) be the set of edges that can be deadheaded at most twice (once) in an optimal solution. These authors reformulate the RPP as a pure binary integer program substituting for each edge $e\in E_{012}$ the integer variable x_e with two binary variables x'_e and x''_e:

$$\text{(5.11)} \qquad x_e=x'_e+x''_e,$$

$$\text{(5.12)} \qquad x'_e, x''_e=0 \quad \text{or} \quad 1.$$

This is equivalent to replacing each edge $e\in E_{012}$ with a pair of parallel edges e' and e''. Let E' (E'') be the set of edges e' (e''), and let $\overline{E}=E_{01}\cup E'\cup E''$. Substituting (5.11) and (5.12) in ($\text{RPP}^{(\text{CORB})}$), Ghiani and Laporte [21] obtain the following formulation:

$$\text{(5.13)} \qquad (\text{RPP}^{(\text{GL1})}) \qquad \text{minimize}\sum_{e\in\overline{E}}c_e x_e$$

$$\text{s.t.} \sum_{e\in\delta(i)} x_e = 0 \bmod 2 \qquad \text{if } i \in V \text{ is } R\text{-even}, \tag{5.14}$$

$$\sum_{e\in\delta(i)} x_e = 1 \bmod 2 \qquad \text{if } i \in V \text{ is } R\text{-odd}, \tag{5.15}$$

$$\sum_{e\in\delta(S)} x_e \geq 2, \qquad S = \cup_{k\in P} V_k, P \subset \{1,\ldots,p\}, P \neq \emptyset, \tag{5.16}$$

$$x_e \in \{0,1\}, \qquad e \in \overline{E}. \tag{5.17}$$

The modulo relations are then removed:

$$(\text{RPP}^{(\text{GL2})}) \qquad \text{minimize} \sum_{e\in\overline{E}} c_e x_e \tag{5.18}$$

$$\text{s.t.} \sum_{e\in\delta(i)\setminus F} x_e \geq \sum_{e\in F} x_e - |F| + 1 \Big(i \in V_R, F \supseteq \delta(i), \tag{5.19}$$

$$|F| \text{ is odd if } i \text{ is R-even }, |F| \text{ is even if } i \text{ is } R\text{-odd} \Big),$$

$$\sum_{e\in\delta(S)} x_e \geq 2, \qquad S = \cup_{k\in P} V_k, P \subset \{1,\ldots,p\}, P \neq \emptyset, \tag{5.20}$$

$$x_e \in \{0,1\}, \qquad e \in \overline{E}. \tag{5.21}$$

Constraints (5.19) are referred to as cocircuit inequalities by Barahona and Grötschel [3]. They state that an even (odd) number of edges are incident to each R-even (R-odd) vertex. Alternatively, constraints (5.19) impose that if an odd (even) number of edges $e \in F$ are incident to a R-even (R-odd) vertex $i \in V$, then at least another edge has to be incident to i. An in-depth analysis of the facets of the polytope of formulation $\text{RPP}^{(\text{GL1})}$ is provided in Reinelt and Theis [34] and Reinelt and Theis [35].

A quite different RRP formulation was proposed by Garfinkel and Webb [18]. These authors construct an augmented graph $G_A(V_A, E_A)$ in which V_A contains all vertices in V, plus a copy i' of each R-even vertex i. The edge set E_A contains an edge for every pair of vertices in V_A except (1) pairs of vertices belonging to the same component and (2) pairs of vertices of which at least one is R-even. The cost $\overline{c}_e$ of an edge $e = (i,j) \in E_A$ is equal to that of a shortest path between vertices i and j in G. Any RPP feasible solution corresponds to a perfect matching in G_A. The connectivity of the solution is ensured by the use of additional real-valued flow variables.

Let $S_1,\ldots,S_p$ be the vertex sets obtained by adding to each set $V_1,\ldots,V_p$ the corresponding copies. Moreover, given a set of vertices $S \subset V_A$, let $\delta_A(S)$ be the set of edges of E_A with one extremity in S and one extremity in $V\setminus S$. Binary variables x_{ij} indicate whether vertex i is matched to vertex j while real-valued y_{ij} variables represent the flow from a vertex i to a vertex j in a different set S_t. The formulation by Garfinkel and Webb [18] is as follows:

$$(\text{RPP}^{(\text{GW})}) \qquad \text{minimize} \sum_{e\in E_A} \overline{c}_e x_e \tag{5.22}$$

$$\text{s.t.} \sum_{e\in\delta_A(i)} x_e = 1, \qquad i \in V_A, \tag{5.23}$$

$$\sum_{i\in S_1}\sum_{j\notin S_1} y_{ij} = p-1, \tag{5.24}$$

$$\sum_{i\notin S_t}\sum_{j\in S_t} (y_{ij}-y_{ji}) = 1, \qquad t = 2,\ldots,p, \tag{5.25}$$

$$y_{ij} \le (p-1)x_e, \qquad i,j \in V_A, i \neq j, e \in E_A : e \in \delta_A(i) \cup \delta_A(i), \tag{5.26}$$

$$x_e \in \{0,1\}, \qquad e \in E_A, \tag{5.27}$$

$$y_{ij} \ge 0, \qquad i,j \in V_A : (i,j) \in E_A, \tag{5.28}$$

$$i \in S_t \text{ for a given } t \in \{1,\ldots,p\} \text{ and } j \notin S_t.$$

In this model, the objective function and constraints (5.23) and (5.27) define a minimum perfect matching in V_A. Constraint (5.24) imposes that S_1 generates $p-1$ units of flow, while constraint (5.25) states that each set of vertices S_t ($t \neq 1$) absorbs one unit of flow. Finally, constraints (5.26) link the matching and flow variables.

Fernández et al. [14] introduced a variant of (RPP$^{(GW)}$) which reduces both the number of flow variables and the number of constraints. Let $w_{tt'}$ be the flow between set S_t ($t = 1,\ldots,p$) and set $S_{t'}$ ($t' = 2,\ldots,p$, $t \neq t'$). The new formulation is

$$(\text{RPP}^{(FMGO)}) \qquad \text{minimize} \sum_{e\in E_A} \bar{c}_e x_e \tag{5.29}$$

$$\text{s.t.} \sum_{e\in\delta_A(i)} x_e = 1, \qquad i \in V_A, \tag{5.30}$$

$$\sum_{j=2,\ldots p} w_{1j} = p-1, \tag{5.31}$$

$$\sum_{t'=1,\ldots,p,t'\neq t} (w_{t't} - w_{tt'}) = 1, \qquad t = 2,\ldots,p, \tag{5.32}$$

$$w_{tt'} \le (p-1) \sum_{e=(i,j)\in E_A : i\in S_t, j\in S_{t'}} x_e, \tag{5.33}$$

$$t = 1,\ldots,p, t' = 2,\ldots,p, t \neq t',$$

$$x_e \in \{0,1\}, \qquad e \in E_A, \tag{5.34}$$

$$w_{tt'} \ge 0, \qquad t = 1,\ldots,p, t' = 2,\ldots,p, t \neq t'. \tag{5.35}$$

As in (RPP$^{(GW)}$), the objective function and constraints (5.30) define a minimum perfect matching in V_A, constraint (5.31) imposes that S_1 generates $p-1$ units of flow, constraints (5.32) state that each set of vertices S_t ($t \neq 1$) absorbs one unit of flow, while constraints (5.33) link the matching and flow variables.

5.4 ▪ Exact algorithms

Christofides et al. [5] developed a lower bound by incorporating constraints (5.2) in the objective function in a Lagrangian fashion. Given a set of multipliers, the relaxation defined by the x variables is equivalent to a shortest spanning tree (SST) problem over $G_C = (V_C, E_C)$. These authors also devised an upper bound by matching the odd-degree vertices in the subgraph induced by the SST and the required edges. These bounding

procedures were then inserted in a branch-and-bound scheme. Twenty-four randomly generated instances, with $|V| \leq 84$, $|A| \leq 184$, and $p \leq 8$, were solved to optimality.

Corberán and Sanchis [9] identified some families of facet-inducing inequalities of formulation (RPP$^{(CORB)}$), including the K-C inequalities (see Section 6.2.2.2) and the following R-odd cut inequalities:

$$\sum_{e \in \delta(S)} x_e \geq 1, \qquad S \subset V, S \text{ is } R\text{-odd}. \tag{5.36}$$

They embedded these inequalities within a cutting plane algorithm in which the separation problems were solved by visual inspection. They were able to solve 23 out of the 24 instances introduced by Christofides et al. [5], as well as two additional instances derived from the street network of Albaida (Spain). It is worth mentioning that polynomial exact separation routines are known for both the connectivity inequalities (5.9) and the R-odd cut constraints (5.36).

Letchford [28] introduced a generalization of the K-C inequalities, called the path-bridge inequalities. The author gives a polynomial-time exact separation routine for a simple subset of the path-bridge inequalities but reports no computational results.

Corberán, Letchford, and Sanchis [6] proposed a cutting plane algorithm for the GRP using connectivity constraints, R-odd cut inequalities, K-C inequalities, path-bridge inequalities, and honeycomb inequalities (see Section 6.2), which can also be applied to the RPP. Since it is based on facet-inducing inequalities, it can yield very strong lower bounds. The algorithm was tested on 118 RPP instances with $16 \leq |V| \leq 116$ and $24 \leq |E| \leq 200$. In 116 cases, optimality was reached with the cuts alone, while the remaining two instances required branching.

Ghiani and Laporte [21] presented an extension,

$$\sum_{e \in \delta(S) \setminus F} x_e \geq \sum_{e \in F} x_e - |F| + 1 \Big(S \subset V_R, F \supseteq \delta(S), \; |F| \text{ is odd if } S \text{ is } R\text{-even}, |F| \text{ is even if } S \text{ is } R\text{-odd} \Big), \tag{5.37}$$

of the cocircuit inequalities (5.19) to a vertex subset S. They also implemented a branch-and-cut algorithm based on the connectivity inequalities (5.20), the R-odd cut inequalities (5.36), and a subset of (5.37),

$$\sum_{e \in \delta(S) \setminus \{f\}} x_e \geq x_f, \qquad S \subset V_R, f \in \delta(S), S \text{ is } R\text{-even}, \tag{5.38}$$

called R-even inequalities. The separation problem for the connectivity inequalities is solved by means of a fast heuristic proposed by Fischetti, Salazar, and Toth [16]. To separate the R-odd cut and R-even cut inequalities, Ghiani and Laporte [21] developed two constructive heuristics. Their algorithm was able to solve to optimality, within moderate solution times, instances with up to 350 randomly generated vertices.

Fernández et al. [14] developed a cutting plane algorithm for their improvement of the formulation of Garfinkel and Webb [18]. Lower bounds are obtained by iteratively solving the LP relaxation of the formulation (RPP$^{(FMGO)}$) enhanced with a set of previously violated connectivity, matching, and K-C inequalities. Violated K-C inequalities are added only when violated inequalities of the other two types have not been found. Upper bounds are provided by a variation of the Frederickson [17] heuristic. The algorithm was able to solve to optimality in 160 out of 173 instances, including the two Albaida instances. For the unsolved instances, the optimality gap was always below 1%.

Corberán and Prins [8] observed that these cutting-plane and branch-and-cut algorithms would certainly be capable of solving larger instances on today's computers because the main limitation was the memory at the time of the experimentations. They also noted that these methods could easily be combined to yield better exact algorithms. Some preliminary results along this direction are reported in Theis [36].

At present the largest Undirected RPP instances ($|V| = 1000$, $|A| \leq 3080$, and $p \leq 205$) were solved by the branch-and-cut procedure for the Windy General Routing Problem presented in Corberán, Plana, and Sanchis [7].

5.5 ▪ Heuristics

One of the most commonly used heuristics for the RPP is the constructive method of Frederickson [17]. Alternative procedures have since been proposed by a number of authors, including Christofides et al. [5], Pearn and Wu [33], Fernández de Córdoba, Garcia Raffi, and Sanchis [15], Hertz, Laporte, and Nanchen Hugo [26], Groves and van Vuuren [25], and Ghiani, Laganà, and Musmanno [20]. In what follows we will briefly describe the first and last four of these contributions, which present the most interest.

5.5.1 ▪ Frederickson's heuristic

Given an undirected graph with p mutually disjoint connected components, Frederickson's heuristic first links these components together by means of an SST. It identifies the odd-degree vertices on the resulting graph and connects them by solving a matching problem, as is done for the undirected CPP [12]. The resulting graph is Eulerian and yields a feasible RPP solution. The steps of the algorithm are as follows:

Step 1. Construct an SST over an auxiliary graph H containing one vertex for each connected component of required edges of G. The cost between any two vertices of H is equal to that of a shortest path between them on G. Let T be the edge set of the spanning tree.

Step 2. Solve a minimum cost matching problem on the set of odd-degree vertices in the graph induced by $E_R \cup T$, where the costs are those of shortest paths on G. Let M be the set of edges induced by the matching.

Step 3. Determine an Eulerian cycle in the graph induced by $E_R \cup T \cup M$.

If E_R is connected, then this algorithm is exact. Otherwise, if the costs c_{ij} satisfy the triangle inequality, it has a worst-case performance ratio of $3/2$. This can be proved by using the same arguments as for the Christofides [4] heuristic for the *Traveling Salesman Problem* (TSP).

Frederickson's heuristic does not always yield an optimal solution because it is not always advantageous to connect the required edges before solving the matching problem (see, for example, Figure 5.1).

5.5.2 ▪ The Monte Carlo method of Fernández de Córdoba, Garcia Raffi, and Sanchis

Fernández de Córdoba, Garcia Raffi, and Sanchis [15] have proposed a rather simple yet effective heuristic based on a probabilistic scheme. Starting at an arbitrary vertex, a tour containing all required edges is iteratively constructed by selecting at each step an edge to

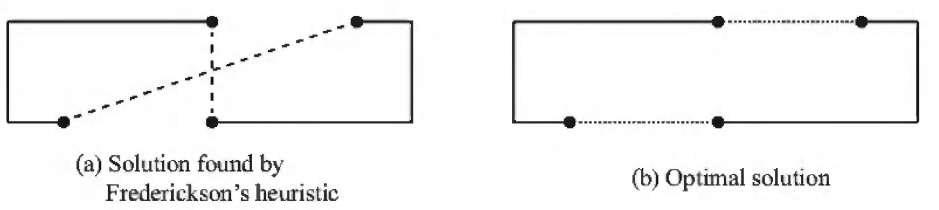


Figure 5.1. *Example for which Frederickson's heuristic does not always yield an optimal solution.*

traverse in a probabilistic manner. When all required edges have been traversed, the tour is closed by returning to the starting vertex through a shortest path.

The probability of selecting an edge depends on its nature and on its cost relative to those of the other edges available for traversal. The authors consider three types of edges: (1) required edges; (2) nonrequired edges incident to a yet untraversed required edge; (3) other nonrequired edges, with decreasing probabilities of selecting edges of type 1, 2, and 3, respectively. Similarly, the probability of selecting an edge is inversely related to its cost. The solution generated by this randomized procedure is then improved by applying three simplification operations: (1) Remove pairs of traversal of the same edge when there are at least three traversals; (2) if a nonrequired edge is traversed twice, remove the two traversals if this does not disconnect the graph; (3) consider a path $(e_1,\ldots,e_k)$ of edges traversed twice between two vertices i and j; then delete one copy of each edge and add one copy of each edge of another path from i to j (e.g., a shortest path).

This heuristic can be repeated a large number of times (1000 times in the authors' implementation). On 26 small and medium-size instances with $7 \leq |V| \leq 102$, $10 \leq |E| \leq 184$, and $4 \leq |E_R| \leq 99$, it identified 19 optimal solutions.

5.5.3 ▪ The improvement heuristics of Hertz, Laporte, and Nanchen Hugo

Hertz, Laporte, and Nanchen Hugo [26] provided a set of improvement heuristics which are applicable to any feasible RPP solution.

The first of these heuristics, called SHORTEN, transforms a solution into an equivalent one starting with a long chain of unrequired edges, which is then shortened by applying a shortest path algorithm. It scans the edges of a solution and applies one of two operations to each edge. If edge (i,j) is required but appears later in the solution as nonrequired, then the operator POSTPONE changes the status of these two edges. If an edge (i,j) is required and does not appear later as nonrequired, then the operator REVERSE attempts to identify a path $(i,j,\ldots,k,i)$ in which edge (k,i) is nonrequired, and it reverses this path. These two procedures are applied as long as it is possible to do so. The longest chain $(i,\ldots,j)$ at the start of the resulting solution is then replaced with a shortest path from i to j.

To illustrate, consider the RPP solution depicted in Figure 5.2(a), where the thick edges are required and the others are not. The statuses of the first and last edges are first reversed. In Figure 5.2(b), the first required edge is (3,4) and this edge does not appear later in the solution. However, the chain (3,4,2,3) can be reversed to yield the solution of Figure 5.2(c). This solution can no longer be transformed by applying POSTPONE or REVERSE. However, it can be shortened by replacing the initial chain (1,2,3,2,4) with the shorter chain (1,2,4), as shown in Figure 5.2(d).

The next two heuristics, called DROP and ADD, are very simple and technical. They are meant to be used as building blocks in other heuristics. DROP removes a required edge (i,j) from the solution. This is achieved by replacing it with a shortest path between i

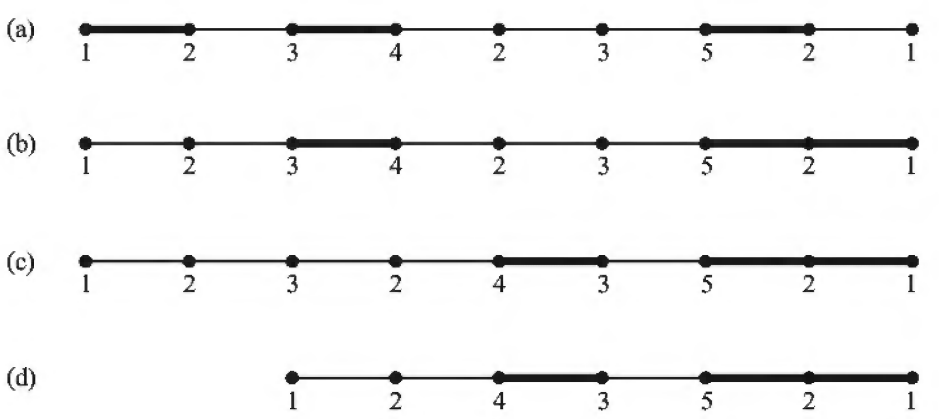


Figure 5.2. *RPP solution.*

and j. ADD inserts a new edge (i,j) into an existing solution by linking i and j to the solution by means of shortest paths.

Using SHORTEN, DROP, and ADD, Hertz, Laporte, and Nanchen Hugo [26] have created the arc routing equivalent of the classical r-opt postoptimization procedures for the TSP [30]. However, the implementation of r-opt in an arc routing context can be rather intricate because required edges are sometimes removed in the process. To illustrate, we restrict our attention to 2-opt. This procedure repeatedly removes two edges from the solution and reconnects the two remaining chains by means of the shortest chains. SHORTEN is then called. If any required edge is missing from the solution because it has been removed, it is reinserted in the solution by means of ADD. The complexity of this heuristic, computed by Groves and van Vuuren [25], is $O(|E|^5)$ per iteration, which makes it unsuitable for large instances.

Another postoptimization procedure is called DROP-ADD: It removes a required edge from the solution, calls SHORTEN, and reinserts the edge by means of ADD. This process is repeated as long as improvements can be obtained.

The algorithms 2-opt and DROP-ADD were tested on several classes of RPP instances. Overall, 2-opt performed better than DROP-ADD but was more time consuming. On some instance classes for which the optimum was known, the latter heuristic yielded an average optimality gap of around 3%, whereas applying 2-opt as a postoptimizer closed this gap to around 0%.

5.5.4 ▪ The heuristics of Groves and van Vuuren

Groves and van Vuuren [25] have designed rather efficient 2-opt and 3-opt heuristics for the undirected RPP. Both operate on a vector $((v_{s_1}, v_{t_1}),\ldots,(v_{s_n}, v_{t_n}))$ whose components are the required edges. A 2-opt (resp., 3-opt) move consists of removing and relocating two (resp., three) components from this vector. A directed layered auxiliary graph $\mathscr{L}$ with vertex set $V(\mathscr{L}) = \{b, s_1t_1, t_1s_1, \ldots, s_nt_n, t_ns_n, e\}$ is then constructed. In this set, b and e are the vertices at which the tour must begin and end. The elements s_it_i and t_is_i of $V(\mathscr{L})$ represent the two traversal directions of edge (v_{s_i}, v_{t_i}). For each $i(1 \leq i \leq n-1)$, an arc of $\mathscr{L}$ is directed from s_it_i to $s_{i+1}t_{i+1}$ and $t_{i+1}s_{i+1}$, and from t_is_i to $s_{i+1}t_{i+1}$ and $t_{i+1}s_{i+1}$. Arcs are also created from b to s_1t_1 and to t_1s_1, and from s_nt_n and t_ns_n to e. Let $d(i,j)$ be the length of a shortest path from vertex i to vertex j in G. Then assign a

cost $d(t_i, s_{i+1})$ (resp., $d(s_i, t_{i+1})$) to the arc $(s_i t_i, s_{i+1} t_{i+1})$ (resp., $(t_i s_i, t_{i+1} s_{i+1})$) for every $i (1 \le i \le n-1)$. Also assign a cost $d(t_i, t_{i+1})$ (resp., $d(s_i, s_{i+1})$) for every $i (1 \le i \le n-1)$. Assign a cost $d(b, s_1)$ (resp., $d(b, t_1)$) to arc $(b, s_1 t_1)$ (resp., $(b, t_1 s_1)$) and a cost $d(t_n, e)$ (resp., $d(s_n, e)$) to arc $(s_n t_n, e)$ (resp., $(t_n s_n, e)$).

Each path from b to e in $\mathscr{L}$ then represents a way of assigning traversal directions to the edges of the vector $((v_{s_1}, v_{t_1}), \ldots, (v_{s_n}, v_{t_n}))$, and a shortest path provides the best traversal directions. The authors show that this shortest path can be computed in $O(n)$ operations. Since such a path would have to be computed at each iteration of a 2-opt or 3-opt algorithm, this would add an order of magnitude to the complexity of each iteration. However, as the authors show, this can be avoided by properly exploiting available information and data structures. Thus, in their implementation, the authors manage to keep the complexity of each iteration of 2-opt and 3-opt to $O(n^2)$ and $O(n^3)$, respectively.

The authors have tested their 2-opt and 3-opt heuristics on instances with Euclidean distances and $507 \le |V| \le 979$, $2528 \le |E| \le 3499$, $304 \le |E_R| \le 350$, and on instances with random distances and $536 \le |V| \le 976$, $2641 \le |E| \le 3451$, $257 \le |E_R| \le 345$. Their results are summarized in Table 5.1.

Table 5.1. *Deviation of the upper bound over the lower bound.*

Data set	Frederickson	2-opt	3-opt
Euclidean	5.9%	3.8%	3.0%
Random	4.1%	2.5%	2.0%

On Euclidean instances with $3130 \le |V| \le 4927$, $20660 \le |E| \le 29835$, $2066 \le |E_R| \le 2984$, the 2-opt heuristic consistently outperformed the Frederickson heuristic within slightly larger computation times.

5.5.5 ▪ The heuristic of Ghiani, Laganà, and Musmanno

Ghiani, Laganà, and Musmanno [20] described a fast heuristic whose use is recommended when a high-quality solution is needed within a short amount of time (e.g., in laser plotter applications). At each iteration, the procedure inserts a connected component of the required edges. A 2-opt local search is then performed in an attempt to reduce the cost. Computational results on a set of benchmark instances with up to 350 vertices show that the algorithm is competitive with the classical Frederickson procedure.

5.6 ▪ Variants

A number of variants and extensions of the undirected RPP, typically related with real-life applications, have been investigated. Two of these, the *Prize-Collecting RPP* [2, 1] and the *Profitable Arc Tour Problem* [13], are presented in Chapter 12. In what follows, we describe variants related to the RPP with multiple edge services, the RPP with deadline classes, and the dynamic RPP.

5.6.1 ▪ The RPP with multiple edge services

We summarize two papers in which edges may have to be serviced several times in a solution. The first, by Ghiani et al. [22], introduces the periodic arc routing problem, in which each required edge $e \in E_R$ must be serviced n_e times over an m-period horizon,

and the periods in which the edges are serviced must be equally spaced. This implies of course that m must be divisible by n_e for all e. This is more restrictive than the conditions imposed in the *Periodic Vehicle Routing Problem* (see, e.g., [11]) in which customers specify allowed combinations of service periods, without requiring equal spacing. Applications of the *Periodic RPP* arise naturally in garbage collection and in street sweeping. The authors have proposed a heuristic in which all edges e having the same service frequency n_e are initially assigned the same service day combination, and a local search is then performed to yield cost reductions. Three procedures are applied: path transfers, cycle transfers, and component transfers.

The first two routines attempt to remove deadheaded edges by suitably moving paths of serviced edges from a service combination to another having the same frequency. The third subroutine verifies, for any service frequency n, whether a cost saving can be obtained by assigning a different service day combination to a connected component induced by the edges with frequency n. In order to assess this heuristic, the authors have generated a number of instances whose optimal solution is known a priori: First, an Eulerian random graph was created; then frequencies were assigned to the edges in such a way that no deadheading would be required. The authors succeeded in solving instances with $|V| = 100, 150, 200, 250$ and $m = 6$. The proposed heuristic yielded optimality gaps varying between 0.58% and 2.92%, depending on the instance classes.

Groves, le Roux, and van Vuuren [24] also solved a variant in which edges may be serviced several times, and a suitable spacing must be achieved between two consecutive services of the same edge. Applications of this problem are encountered in the maintenance of transportation networks and in garbage collection or snow plowing operations. The two goals of cost minimization and of service spacing are somewhat conflictual. They were handled simultaneously through a bicriteria objective. The authors have considered two ways of modeling the spacing constraint: through the number of edge traversals between consecutive services of the same edge, and through a temporal deviation within a specified time window. The second of these options proved to be the most satisfactory.

The authors developed a simple construction heuristic which links circuit segments through several copies of the graph. Local search moves, such as those described in Groves and van Vuuren [25], are then applied. Instances defined on six types of graphs, with $20 \leq |E| \leq 298$, were solved. Since no good lower bounds are available for this problem, it is difficult to assess the accuracy of this heuristic. It was observed that the average improvement yielded by the local search phase over the construction phase was equal to 13%.

5.6.2 ▪ The RPP with deadline classes

In the RPP with deadline classes [29], the set E_R of required edges is partitioned into $\{E_R^1, \ldots, E_R^p\}$ and service for each class k of edges ($k = 1, \ldots, p$) must be completed by time T^k. The authors have developed an integer linear programming model for this problem. They have also proposed several classes of valid inequalities which exploit the structure of the problem, but they have not attempted to identify conditions under which these induce facets. The valid inequalities are called strong cumulative inequalities, strong cumulative connectivity inequalities, strong cumulative parity inequalities, and strong cumulative path-bridge inequalities. Some of these inequalities are natural counterparts of similar inequalities derived for the RPP [9, 10, 6, 28]. The authors have also provided lifted versions of these inequalities. The connectivity and parity inequalities can be separated in polynomial time, while heuristics are used for the strong cumulative path-bridge

inequalities. The problem was solved by branch-and-cut. Instances with $17 \leq |V| \leq 50$, $35 \leq |E| \leq 110$, $22 \leq |E_R| \leq 67$, and $p = 1$ or 2 were solved to optimality.

5.6.3 ▪ The dynamic RPP

The dynamic RPP investigated by Moreira et al. [31] is motivated by an industrial application in which pieces of different shapes have to be cut out of a circular area by means of an electrified copper string. In a first phase, the pieces are nested (or packed) in the circular area (Figure 5.3). In a second phase, which is relevant as far as the RPP is concerned, the pieces are cut out by the copper string and fall out of the circle when their entire perimeter has been cut. Any movement of the cutting tool generates an effective cut, unless of course it is performed over a region already cut out. Like the laser-beam application described by Ghiani and Improta [19], this problem can be modeled as an RPP. However, because pieces are sometimes cut out, the graph on which the RPP is solved changes in a dynamic fashion (Figure 5.4). Also, because some of the pieces may share common edges, other practical considerations complicate the problem; namely, there may not remain uncut pieces in areas that have been cut out.

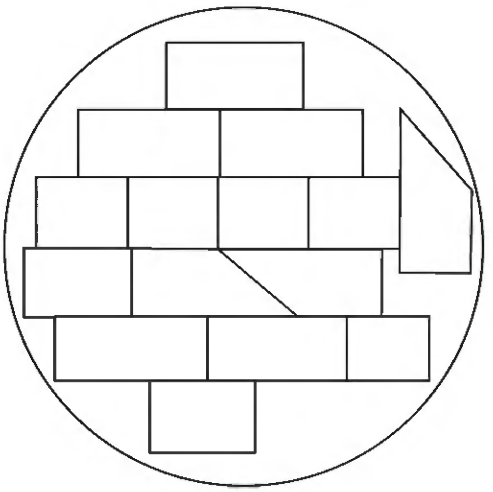
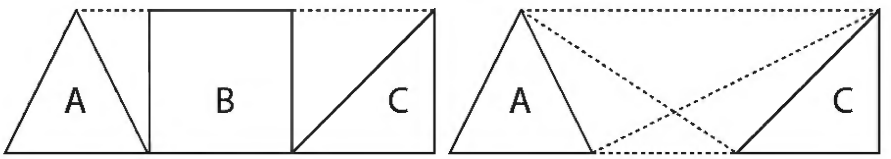


Figure 5.3. *Pieces nested in a circular area.*



(a) Before B has been cut out (b) After B has been cut out

Figure 5.4. *RPP graph before and after piece B has been cut out.*

The authors have devised two constructive heuristics for this problem. The first, called "higher up vertex" (HUV): Given the current vertex, is a higher up vertex among the vertices reachable by uncut edges, under some side conditions? The second, called "minimum empty path" (MEP) is a variant of HUV which uses different criteria for the

execution of unrequired movements. The two heuristics were tested and compared on 10 industrial instances containing between 19 and 52 pieces, and between 183 and 639 vertices. Deviations from a lower bound were computed. On average, HUV yielded an optimality gap of 17.5%, whereas this gap was equal to 15.2% for MEP.

Bibliography

[1] J. ARÁOZ, E. FERNÁNDEZ, AND O. MEZA, *Solving the prize-collecting rural postman problem*, European Journal of Operational Research, 196 (2009), pp. 886–896.

[2] J. ARÁOZ, E. FERNÁNDEZ, AND C. ZOLTAN, *Privatized rural postman problems*, European Journal of Operational Research, 33 (2006), pp. 3432–3449.

[3] F. BARAHONA AND M. GRÖTSCHEL, *On the cycle polytope of a binary matroid*, Journal of Combinatorial Theory, 40 (1986), pp. 40–62.

[4] N. CHRISTOFIDES, *Worst-case analysis of a new heuristic for the travelling salesman problem*, Tech. Report 388, School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA, 1976.

[5] N. CHRISTOFIDES, V. CAMPOS, Á. CORBERÁN, AND E. MOTA, *An algorithm for the rural postman problem*, Tech. Report IC.O.R.81.5, Imperial College, London, 1981.

[6] Á. CORBERÁN, A. N. LETCHFORD, AND J. M. SANCHIS, *A cutting plane algorithm for the general routing problem*, Mathematical Programming, Series A, 90 (2001), pp. 291–316.

[7] Á. CORBERÁN, I. PLANA, AND J. M. SANCHIS, *A branch & cut algorithm for the windy general routing problem and special cases*, Networks, 49 (2007), pp. 245–257.

[8] Á. CORBERÁN AND C. PRINS, *Recent results on arc routing problems: An annotated bibliography*, Networks, 56 (2010), pp. 50–69.

[9] Á. CORBERÁN AND J. M. SANCHIS, *A polyhedral approach for the rural postman problem*, European Journal of Operational Research, 79 (1994), pp. 95–114.

[10] ——, *The general routing problem polyhedron: Facets from the RPP and GTSP polyhedra*, European Journal of Operational Research, 108 (1998), pp. 538–550.

[11] J.-F. CORDEAU, M. GENDREAU, AND G. LAPORTE, *A tabu search heuristic for periodic and multi-depot vehicle routing problems*, Networks, 30 (1997), pp. 105–119.

[12] J. EDMONDS AND E. L. JOHNSON, *Matching, Euler tours and the Chinese postman problem*, Mathematical Programming, 5 (1973), pp. 88–124.

[13] D. FEILLET, P. DEJAX, AND M. GENDREAU, *The profitable arc tour problem: Solution with a branch and price algorithm*, Transportation Science, 39 (2005), pp. 539–552.

[14] E. FERNÁNDEZ, O. MEZA, R. GARFINKEL, AND M. ORTEGA, *On the undirected rural postman problem: Tight bounds based on a new formulation*, Operations Research, 51 (2003), pp. 281–291.

[15] P. FERNÁNDEZ DE CÓRDOBA, L. M. GARCIA RAFFI, AND J. M. SANCHIS, *A heuristic algorithm based on Monte Carlo methods for the rural postman problem*, Computers & Operations Research, 25 (1998), pp. 1097–1105.

[16] M. FISCHETTI, J. J. SALAZAR, AND P. TOTH, *A branch-and-cut algorithm for the symmetric generalized travelling salesman problem*, Operations Research, 45 (1997), pp. 378–393.

[17] G. N. FREDERICKSON, M. S. HECHT, AND C. E. KIM, *Approximation algorithms for some routing problems*, SIAM Journal on Computing, 7 (1978), pp. 178–193.

[18] R. GARFINKEL AND I. WEBB, *On crossings, the crossing postman problem and the rural postman problem*, Networks, 34 (1999), pp. 173–180.

[19] G. GHIANI AND G. IMPROTA, *The laser-plotter beam routing problem*, The Journal of the Operational Research Society, 52 (2001), pp. 887–903.

[20] G. GHIANI, D. LAGANÀ, AND R. MUSMANNO, *A constructive heuristic for the undirected rural postman problem*, Computers & Operations Research, 33 (2006), pp. 3450–3457.

[21] G. GHIANI AND G. LAPORTE, *A branch-and-cut algorithm for the undirected rural postman problem*, Mathematical Programming, 87 (2000), pp. 467–481.

[22] G. GHIANI, R. MUSMANNO, G. PALETTA, AND C. TRIKI, *A heuristic for the periodic rural postman problem*, Computers & Operations Research, 32 (2005), pp. 219–228.

[23] M. GRÖTSCHEL, M. JÜNGER, AND G. REINELT, *Optimal control of plotting and drilling machines: A case study*, Zeitschrift für Operations Research, 35 (1991), pp. 61–84.

[24] G. W. GROVES, J. LE ROUX, AND J. H. VAN VUUREN, *On a routing and scheduling problem concerning multiple edge traversals in graphs*, Networks, 46 (2005), pp. 69–81.

[25] G. W. GROVES AND J. H. VAN VUUREN, *Efficient heuristics for the rural postman problem*, ORiON, 21 (2005), pp. 33–51.

[26] A. HERTZ, G. LAPORTE, AND P. NANCHEN HUGO, *Improvement procedures for the undirected rural postman problem*, INFORMS Journal on Computing, 11 (1999), pp. 53–62.

[27] J. K. LENSTRA AND A. H. G. RINNOOY KAN, *On general routing problems*, Networks, 6 (1976), pp. 273–280.

[28] A. N. LETCHFORD, *New inequalities for the general routing problem*, European Journal of Operational Research, 96 (1997), pp. 317–322.

[29] A. N. LETCHFORD AND R. W. EGLESE, *The rural postman problem with deadline classes*, European Journal of Operational Research, 105 (1998), pp. 390–400.

[30] S. LIN, *Computer solutions of the traveling salesman problem*, Bell System Technical Journal, 44 (1965), pp. 2245–2269.

[31] L. M. MOREIRA, J. F. OLIVEIRA, A. M. GOMEZ, AND J. S. FERREIRA, *Heuristics for a dynamic rural postman problem*, Computers & Operations Research, 34 (2007), pp. 3281–3294.

[32] C. S. ORLOFF, *A fundamental theorem in vehicle routing*, Networks, 27 (1974), pp. 35–64.

[33] W. L. PEARN AND T. C. WU, *Algorithms for the rural postman problem*, Computers & Operations Research, 22 (1995), pp. 819–828.

[34] G. REINELT AND D. O. THEIS, *Transformation of facets of the general routing problem polytope*, SIAM Journal on Optimization, 16 (2005), pp. 220–234.

[35] ——, *On the general routing polytope*, Discrete Applied Mathematics, 156 (2008), pp. 368–384.

[36] D. O. THEIS, *Das General Routing Problem mit binären Variablen*, Ph.D. thesis, Diplomarbeit, Ruprecht-Karls-Universität Heidelberg, Institut für Informatik, Heidelberg, Germany, 2001.

Chapter 6

The Rural Postman Problem on Directed, Mixed, and Windy Graphs

Ángel Corberán
Isaac Plana
José María Sanchis

6.1 ▪ Introduction

The previous chapter provided an overview of the *Rural Postman Problem* (RPP) on an undirected graph. Recall that given a graph and a subset of required edges, this problem consists of finding a minimum cost tour (a closed walk) traversing every required edge at least once. This problem was proposed by Orloff [49] and shown to be NP-hard by Lenstra and Rinnooy Kan [45]. This chapter deals with its directed, mixed, and windy versions. For simplicity, and given that here we will work with directed, mixed, and windy graphs, we will use the term "link" to refer to (undirected) edges and (directed) arcs indistinctly.

A problem closely related to the RPP is the *General Routing Problem* (GRP). Given a graph with nonnegative costs associated with its links, a subset of required links, and a subset of required vertices, the GRP consists of finding a minimum cost tour traversing each required link at least once and visiting each required vertex at least once. The GRP is a generalization of the RPP, so many of the results obtained on the GRP can be applied to the RPP. Since it is not as widely known as the RPP and it is not a pure arc routing problem, the GRP will not be addressed in this chapter, but the results found in several studies on the GRP defined on directed, mixed, and windy graphs will be referenced and applied to the corresponding version of the RPP.

The *Mixed Rural Postman Problem* (MRPP), defined over a mixed graph (containing arcs and edges), is an NP-hard problem (Corberán, Romero, and Sanchis [27]), since it contains the RPP as a particular case when there are no arcs in the graph. In the MRPP, we have a subset of required edges and a subset of required arcs. It can be assumed that all the vertices of the graph are incident with at least one required link. If this is not the case, the instance can easily be transformed into an equivalent one in which this property holds (see e.g. [27]).

When the RPP is defined on a directed graph (containing only arcs), we have the *Di-*

rected Rural Postman Problem (DRPP), which is also NP-hard (Lenstra and Rinnooy Kan [45]). This problem will be studied at the end of this chapter.

The *Windy Rural Postman Problem* (WRPP) is defined on an undirected graph where each edge has two associated costs, one for each direction of traversal. This problem also contains the RPP as a special case and therefore is NP-hard too (Benavent et al. [9] and [8]). As with the MRPP, we can assume that all the vertices in the graph are incident with at least one required edge. The study of the WRPP is of special interest for two reasons. First, it generalizes the RPP on undirected, directed, and mixed graphs. Secondly, while the MRPP has obvious applications in the context of garbage collection, mail delivery, etc., there are several other specific real-life applications that can be modeled using windy graphs only. Two of these applications are described in what follows.

The periodical inspection of complex structures, such as bridges or skyscrapers, requires checking the integrity of the surface of the beams in their skeleton (see Figure 6.1). Since this is a risky task, it could be carried out by remotely controlled robots equipped with a camera, such as the one in Figure 6.2 (Balaguer et al. [7]). The beam structure can be represented by means of an undirected graph where some edges representing the surfaces of the beams (dashed lines in the graph in Figure 6.1) are required, while some others representing the movements needed to move from one surface to another (bold and dotted lines in the graph in Figure 6.1) are not required. Moreover, each edge has two associated costs corresponding to the amount of energy required to perform this movement in each possible direction. Note that these two costs can be different, given that gravity affects energy consumption. Therefore, the problem of designing a route that traverses all the required edges at minimum cost is a WRPP (Benavent et al. [8]).

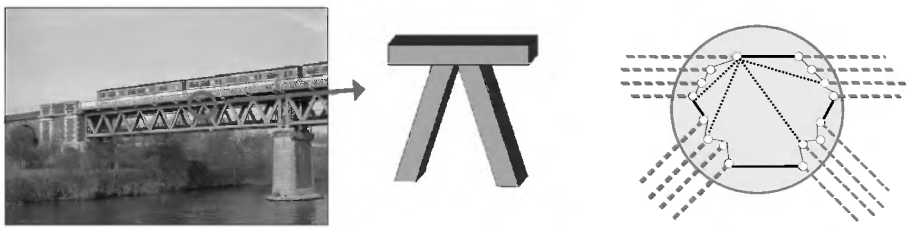


Figure 6.1. *Bridge, beam structure, and graph modeling [8].*

A further application of the WRPP is related to the design of the movements of cutting plotters, such as that in Figure 6.3. These machines are used to print and cut signs, stickers, etc. To this end, the machine is equipped with a razor that can be moved in two directions. The objective is to design the movements of the paper and the razor in order to complete the cutting of all the designs in the minimum possible time. Moreover, the cost of moving the paper in one direction is different from that of moving it in the opposite one, and therefore we may have nonsymmetric costs for the edges. The movements that represent the cutting of the paper are modeled as required edges, while the movements of the paper and the razor that do not require cutting correspond to nonrequired edges.

6.2 ▪ The mixed Rural Postman Problem

In this section, a formal definition of the MRPP is given, some notation is introduced, and heuristic and exact algorithms for this problem are presented.

Consider a strongly connected mixed graph $G = (V, E, A)$ with a vertex set V, an edge set E, and an arc set A, and a nonnegative cost c_e for each $e \in E \cup A$. Given a subset

Figure 6.2. *A climber Robot.*

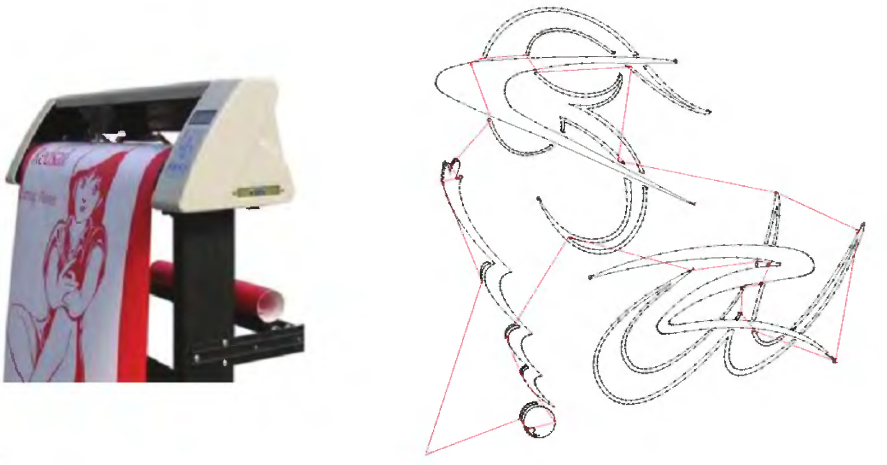


Figure 6.3. *Cutting plotter and design fragment (provided by S. Defever).*

of required edges $E_R \subseteq E$ and a subset of required arcs $A_R \subseteq A$, the MRPP consists of finding a minimum cost tour traversing all the required edges and all the required arcs at least once.

In what follows, it will be assumed that all the edges in the graph are required, i.e., $E_R = E$, since the original graph can easily be transformed into another one satisfying this property (see e.g. Christofides et al. [16]) by replacing each nonrequired edge with two opposite nonrequired arcs of the same cost. Note that the graph induced by the required links $G(V, E \cup A_R)$ is in general nonconnected. The vertex sets of its connected components, denoted by $V_1, \ldots, V_p$, are called R-sets.

6.2.1 ▪ Problem formulations

The MRPP, just as the *Mixed Chinese Postman Problem* (MCPP) and other ARPs defined on mixed graphs, can be formulated in two ways by using two or just one variable per edge of the graph. Both formulations have been widely used in the literature and compared in Corberán, Mota, and Sanchis [22] (see also Chapter 4). However, these formulations and

their associated polyhedra have not been studied directly in the literature for the MRPP. Formulation F1 has been studied for the *Mixed General Routing Problem* (MGRP), in Corberán, Romero, and Sanchis [27] and Corberán, Mejía, and Sanchis [21], while formulation F2 is obtained as a particular case of the *Windy General Routing Problem* (WGRP), which has been studied in Plana [50] and Corberán, Plana, and Sanchis [25, 26].

6.2.1.1 ▪ Formulation F1

Let us define x_e as the number of times a link $e \in E \cup A$ appears in the tour. If it is necessary to make an explicit reference to the direction in which a required link is traversed, x_{ij} will be used instead of x_e.

For $S_1, S_2 \subseteq V$, $(S_1 : S_2)$ denotes the set of links with one end-point in S_1 and the other in S_2, $A(S_1 : S_2) = \{(i,j) \in A : i \in S_1, j \in S_2\}$, and $E(S_1 : S_2)$ denotes the set of edges with one endpoint in S_1 and the other one in S_2. Moreover, $\delta^+(S) = A(S : V \setminus S)$, $\delta^-(S) = A(V \setminus S : S)$, and $\delta(S) = E(S : V \setminus S)$, while $E(S)$ and $A(S)$ denote the sets of edges and arcs, respectively, with both endpoints in S. Furthermore, $\delta^*(S) = \delta(S) \cup \delta^+(S) \cup \delta^-(S) = (S : V \setminus S)$. In order to denote the required links of a given set, subindex R will be used (e.g., $(S_1 : S_2)_R$ represents the set of required links between S_1 and S_2). Finally, for a subset of links F, $x(F) = \sum_{e \in F} x_e$.

A mixed graph $G = (V, E, A)$ is called *even* if all its vertices have even degree and *balanced* if, given any subset S of vertices, the difference between the number of arcs directed from S to $V \setminus S$, $|\delta^+(S)|$, and the number of arcs directed from $V \setminus S$ to S, $|\delta^-(S)|$, is no greater than the number of edges joining S and $V \setminus S$, $|\delta(S)|$.

Thus, the MRPP can be formulated as

$$\text{(6.1)} \quad \text{(MRPP-F1)} \quad \min \sum_{e \in E \cup A} c_e x_e$$

$$\text{(6.2)} \quad \text{s.t.} \quad x(\delta^*(i)) \equiv 0 \bmod 2 \qquad \forall i \in V,$$

$$\text{(6.3)} \quad x(\delta^+(S)) \geq 1 \qquad \forall S = \cup_{k \in Q} V_k,\ Q \subset \{1, \ldots, p\},$$

$$\text{(6.4)} \quad x(\delta^+(S)) - x(\delta^-(S)) \leq x(\delta(S)) \qquad \forall S \subset V,$$

$$\text{(6.5)} \quad x_e \geq 1 \text{ and integer} \qquad \forall e \in E \cup A_R,$$

$$\text{(6.6)} \quad x_e \geq 0 \text{ and integer} \qquad \forall e \in A \setminus A_R.$$

The above formulation is stated in terms of tours, where each variable represents the number of times the associated link is traversed, while the one given in Corberán, Romero, and Sanchis [27] was presented using what the authors called semitours, which are the result of removing one copy of each required link from a tour. In what follows, the inequalities will be expressed considering the solutions as tours.

As with the MCPP (see Chapter 4), if $K_1, \ldots, K_q$ denote the sets of vertices of the connected components of the graph (V, E), all the MRPP solutions satisfy the following q system equations:

$$\text{(6.7)} \qquad x(\delta^+(K_i)) = x(\delta^-(K_i)), \quad i = 1, \ldots, q,$$

of which $q - 1$ are linearly independent.

6.2.1.2 ▪ MRPP polyhedron associated with formulation F1

Let $\text{MRPP}_{F1}(G)$ be the convex hull of all the tours $x \in \mathbb{R}^{E \cup A}$ satisfying (6.2) to (6.6). $\text{MRPP}_{F1}(G)$ is an unbounded polyhedron with $\dim(\text{MRPP}_{F1}(G)) = |E \cup A| - q + 1$ and

the following inequalities included in the formulation of the MRPP are facet-inducing under mild conditions (Corberán, Romero, and Sanchis [27]):

- Trivial inequalities: $x_e \geq 1 \ \forall e \in E \cup A_R$ and $x_e \geq 0 \ \forall e \in A \setminus A_R$.
- Connectivity inequalities (6.3).
- Balanced-set inequalities (6.4).

In what follows, other inequalities defining facets of $\text{MRPP}_{F1}(G)$ are presented.

R-odd cut inequalities

A cutset $\delta^*(S)$ is called R-odd if it contains an odd number of required links. The following R-odd cut inequalities were shown to be valid and facet-inducing for $\text{MRPP}_{F1}(G)$ in Corberán, Romero, and Sanchis [27]:

$$x(\delta^*(S)) \geq |\delta^*_R(S)| + 1, \ \forall \delta^*(S) \quad R\text{-odd cutset of } G. \tag{6.8}$$

K-C and K-C_{02} inequalities

K-C inequalities were proposed for the undirected RPP (Corberán and Sanchis [28]) and generalized to the MRPP in Corberán, Romero, and Sanchis [27]. Let K be an integer with $K \geq 3$. Let us consider a partition of V into $K+1$ subsets $\{M_0, M_1, \ldots, M_K\}$ such that each R-set V_i, $1 \leq i \leq p$, is contained in exactly one of the node sets $M_0 \cup M_K, M_1, \ldots, M_{K-1}$ and $(M_0 : M_K)$ contains a positive and even number of required links. Given a link $(i,j) \in E \cup A$ with $i \in S$, $j \in T$, the coefficient corresponding to its associated variable is $c(S,T)$, where $c(M_0, M_K) = K-2$ and $c(M_i, M_j) = |i-j|$, $\forall i,j : \{i,j\} \neq \{0,K\}$ (see Figure 6.4(a)).

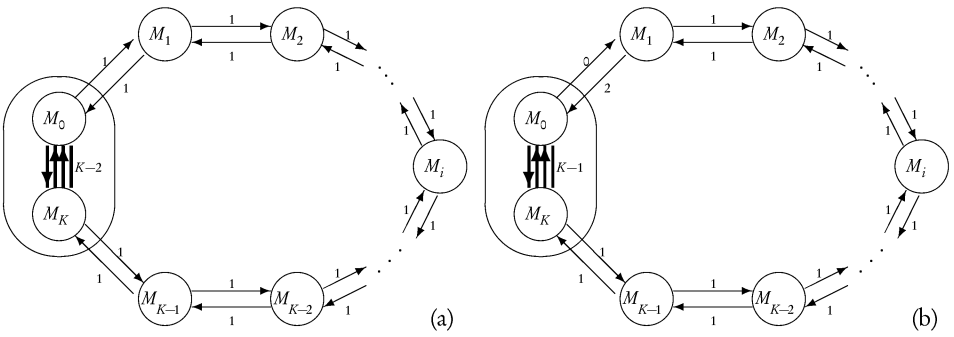


Figure 6.4. *K-C and K-C_{02} configurations.*

The coefficients associated with variables corresponding to links joining vertices of the same subset M_i are zero. The K-C inequality is

$$(K-2)\, x\,(M_0 : M_K) + \sum_{\substack{0 \leq i < j \leq K \\ (i,j) \neq (0,K)}} |i-j|\, x(M_i : M_j) \geq 2(K-1) + (K-2)|(M_0 : M_K)_R|. \tag{6.9}$$

These inequalities are valid for the MRPP, and they are facet-inducing for $\text{MRPP}_{F1}(G)$ if the induced subgraphs $G(M_i)$, $i = 0,1,\ldots,K$, are strongly connected, the sets $A(M_i : M_{i+1})$ and $A(M_{i+1} : M_i)$ are nonempty, and

$$|E(M_0 : M_K)| \geq \big||A_R(M_K : M_0)| - |A_R(M_0 : M_K)|\big|,$$

i.e., if there are enough edges in $(M_0 : M_K)$ to balance this cutset.

K-C inequalities have symmetric coefficients, i.e., $c(S,T) = c(T,S)$, because they were obtained from those proposed for the undirected RPP. Therefore, they ignore the directed nature of the MRPP, where we can have opposite arcs with different costs. The following inequalities, called K-C_{02}, try to exploit this fact by using different coefficients for some opposite arcs. They are based on the same partition of V, with the only difference being that now K can be 2. Figure 6.4(b) shows the coefficients of the links joining consecutive sets. Arcs in $A(M_i : M_j)$ not represented in the figure have a coefficient equal to the length of the shortest path from M_i to M_j. Note that the links in $(M_0 : M_K)$ now have a coefficient $K-1$, while the opposite arcs in $(M_0 : M_1)$ now have coefficients 0 and 2. In fact, these 0-2 coefficients could be placed between any pair of consecutive subsets M_i, M_{i+1}, but the corresponding inequalities would induce the same facets. If we denote the coefficient associated with each link e by α_e, the K-C_{02} inequality is

$$\sum_{e \in E \cup A} \alpha_e x_e \geq 2(K-1) + (K-1)|(M_0 : M_K)_R|. \tag{6.10}$$

In Corberán, Romero, and Sanchis [27] it is proved that these inequalities are valid and facet-inducing for $\text{MRPP}_{F1}(G)$ under certain conditions.

Path-Bridge and Path-Bridge$_{02}$ inequalities

In the K-C and K-C_{02} inequalities there is a single path of sets $M_1,\ldots,M_{K-1}$ joining sets M_0 and M_K. The generalizations of these inequalities in which any number of paths is considered are the Path-Bridge and Path-Bridge$_{02}$ inequalities. Path-Bridge inequalities were introduced by Letchford [46] for the undirected GRP and generalized to the MGRP in Corberán, Romero, and Sanchis [27].

Let $P \geq 1$ and $B \geq 0$ be two integers such that $P + B \geq 3$ and odd, and let $n_i \geq 2$ be integers, $i = 1,\ldots,P$. Let us consider a partition of V into subsets $\{A, Z, M_j^i : i = 1,\ldots P,\ j = 1,\ldots n_i\}$ (where, for convenience, for all i we identify M_0^i with A and $M_{n_i+1}^i$ with Z) satisfying that each R-set V_i is contained in exactly one of the node sets $A \cup Z$ or M_j^i (i.e., each required link either lies in certain $E(M_j^i) \cup A(M_j^i)$ or crosses from A to Z) and $(A : Z)$ contains a number B of required links. Given a link $e = (i,j) \in E \cup A$ with $i \in S$, $j \in T$, the coefficient corresponding to its associated variable is $\alpha_e = c(S,T)$, where

- $c(A,Z) = c(Z,A) = 1$,
- $c(M_j^i, M_q^i) = \dfrac{|j-q|}{n_i - 1} \qquad \forall j,q \in \{0,1,\ldots,n_i+1\}, 0 < |j-q| < n_i + 1,$
- $c(M_j^i, M_q^r) = \dfrac{1}{n_i - 1} + \dfrac{1}{n_r - 1} + \left|\dfrac{j-1}{n_i - 1} - \dfrac{q-1}{n_r - 1}\right|$
  $\forall i,r \in \{1,\ldots,P\},\ i \neq r,\ \ j \in \{1,\ldots,n_i\},\ q \in \{1,\ldots,n_r\}.$

Again, coefficients associated with variables corresponding to links joining vertices in the same set M_j^i are zero. Some of these coefficients are shown in Figure 6.5. The Path-Bridge inequality is then

$$\sum_{e\in E\cup A} \alpha_e x_e \geq B+1+\sum_{i=1}^{P} \frac{n_i+1}{n_i-1}+|(A:Z)_R|. \tag{6.11}$$

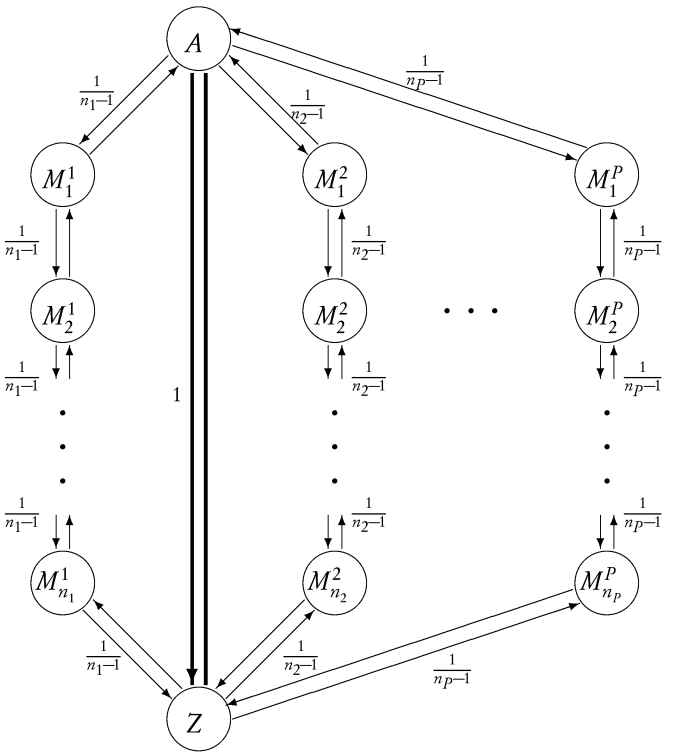


Figure 6.5. *Coefficients of a Path-Bridge inequality.*

In the special case in which n_i is the same number n for each path, these inequalities are called regular Path-Bridge inequalities and can be rewritten in a simpler form where all the coefficients are integer by multiplying them by $n-1$.

Path-Bridge inequalities are valid for $\text{MRPP}_{F1}(G)$. Provided that all the induced graphs $G(M_i)$ are strongly connected and all the sets $A(M_j^i : M_{j+1}^i)$ and $A(M_{j+1}^i : M_j^i)$, $i = 1,\ldots,P$, $j = 0,1,\ldots,n_i$, are nonempty, they are also facet-inducing for $\text{MRPP}_{F1}(G)$ under the following conditions:

$$|E(A:Z)| \geq \big||A_R(A:Z)|-|A_R(Z:A)|\big| \text{ if } P \text{ is odd, or}$$

$$|E(A:Z)| \geq \big||A_R(A:Z)|-|A_R(Z:A)|\big|+1 \text{ if } P \text{ is even.}$$

As happens with K-C inequalities, there is an asymmetric version of the Path-Bridge inequalities, called Path-Bridge$_{02}$, that are valid and facet-inducing for $\text{MRPP}_{F1}(G)$ (see Corberán, Romero, and Sanchis [27] for details).

Honeycomb and Honeycomb$_{02}$ inequalities

The Honeycomb inequalities were introduced in Corberán and Sanchis [29] for the undirected GRP and in Corberán, Mejía, and Sanchis [21] for the MRPP. Like Path-Bridge inequalities, they are also a generalization of K-C inequalities. However, the generalization is in a different direction and neither class contains the other. In the partition associated with a K-C inequality, an R-set (or a cluster of R-sets) is divided into two parts (M_0 and M_K). Now this partition is generalized simultaneously both in the number of parts an R-set is divided into and in the number of divided R-sets.

Consider a partition of the set of vertices V into K vertex sets $\{A_1,\ldots,A_L,A_{L+1},\ldots,A_K\}$, $K \geq 3$, in such a way that each R-set is contained in exactly one A_i and the induced subgraphs $G(A_i)$, $i = 1,\ldots,K$, are strongly connected. Suppose each set A_i, $i = 1,\ldots,L$, can be partitioned into $\gamma_i \geq 2$ subsets, $A_i = B_i^1 \cup B_i^2 \cup \ldots \cup B_i^{\gamma_i}$, satisfying the following conditions:

1. $|\delta_R^*(B_i^p)|$ is even for $p = 1,\ldots,\gamma_i$.
2. The induced subgraphs $G(B_i^p)$, $p = 1,\ldots,\gamma_i$, are strongly connected.
3. The graph with vertex set $B_i^1,\ldots,B_i^{\gamma_i}$ and having an edge (B_i^p,B_i^q) for every pair of vertices $B_i^p \neq B_i^q$, such that $(B_i^p : B_i^q)_R \neq \emptyset$ in the original graph, is connected.

For notational convenience, $B_i^0 = A_i$, $i = L+1,\ldots,K$. We therefore have the following partition of V:

$$\mathscr{B} = \{B_1^1,\ldots,B_1^{\gamma_1},B_2^1,\ldots,B_2^{\gamma_2},\ \ldots\ ,B_L^1,\ldots,B_L^{\gamma_L},\ B_{L+1}^0,\ldots,B_K^0\}.$$

Let $G_{\mathscr{C}} = (\mathscr{B},\mathscr{E} \cup \mathscr{A})$ be the graph defined by this partition, where the set of links $\mathscr{E} \cup \mathscr{A}$ contains a required link (B_i^p,B_j^q) for each required link between sets B_i^p and B_j^q in the original graph and a nonrequired arc (B_i^p,B_j^q) between each couple of vertices B_i^p and B_j^q such that there is a nonrequired arc from B_i^p to B_j^q in the original graph.

Let us suppose that there is a set T of pairs of opposite nonrequired arcs in $G_{\mathscr{C}}$ joining vertices corresponding to different A_i, $i = 1,\ldots K$, such that the undirected graph with vertex set $\mathscr{B}$ and having an edge (B_i^p,B_j^q) for each pair of opposite arcs (B_i^p,B_j^q), (B_j^q,B_i^p) in T is a spanning tree. For each pair of vertices B_i^p, B_j^q in $\mathscr{B}$, let $d(B_i^p,B_j^q)$ denote the number of arcs in the unique path in $(\mathscr{B},T)$ joining B_i^p to B_j^q. Assume that the following two conditions are also satisfied:

4. $d(B_i^p,B_i^q) \geq 3 \quad \forall\ i = 1,\ldots,L$ and $\forall\ p \neq q$.
5. The degree of every vertex B_i^p, $p \neq 0$, in $(\mathscr{B},T)$ is equal to 1.

The graph $(\mathscr{B},T)$ can be seen in Figure 6.6, where the arcs in T are represented by thin lines and the required links by bold lines and it looks like a honeycomb where each cell is a K-C partition defined by a pair of vertices B_i^p, B_i^q (related to the same A_i) and by the unique path in T joining B_i^p and B_i^q.

The coefficients of the variables corresponding to the links in the original graph in a Honeycomb inequality are as follows (see Figure 6.6):

- For the links $e \in (B_i^p : B_i^q)$ with $p,q \neq 0$ (the nodes obtained by splitting the sets A_i, $i = 1,\ldots,L$), $\alpha_e = d(B_i^p,B_i^q) - 2$.

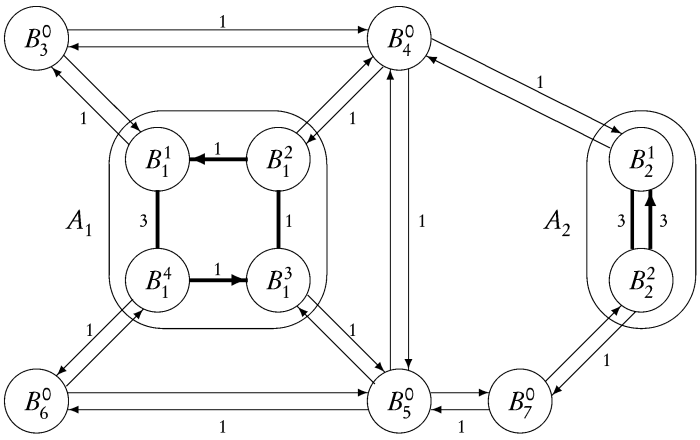


Figure 6.6. *Honeycomb partition.*

- For the arcs $e \in A(B_i^p : B_j^q)$, $i \neq j$, $\alpha_e = d(B_i^p, B_j^q)$.

- For the remaining links, $\alpha_e = 0$.

Then the following inequality is called a Honeycomb inequality and is valid for the MRPP and is facet-inducing under certain conditions:

$$\sum_{e \in E \cup A} \alpha_e x_e \geq 2(K-1) + \sum_{e \in \mathscr{R}} \alpha_e, \tag{6.12}$$

where $\mathscr{R}$ is the set of required links joining two distinct sets B_i^p and B_i^q with $p, q \neq 0$. For example, the right-hand side of the Honeycomb inequality represented in Figure 6.6 is $2 \times (7-1) + 3 + 1 + 1 + 1 + 3 + 3 = 24$.

In Corberán, Mejía, and Sanchis [21], a more general version of the Honeycomb inequalities (6.12) is presented by ignoring condition 5, but in this case some of the coefficients in the inequality have to be calculated by means of a sequential lifting procedure.

As happens with K-C and Path-Bridge inequalities, there is an asymmetric version of the Honeycomb inequalities. They are called Honeycomb$_{02}$, but the only case which has been studied is the one where one R-set is partitioned. These inequalities are valid and facet-inducing for MRPP$_{F1}(G)$ (see [21] for details).

6.2.1.3 ▪ Formulation F2

Instead of using just one variable representing each edge, the MRPP can also be formulated using two variables x_{ij} and x_{ji} associated with each edge $e = (i, j)$, representing the number of times edge e is traversed in the corresponding direction. Then the MRPP can be formulated as

$$\text{(6.13)}\quad \text{(MRPP-F2)}\quad \min \sum_{e=(i,j)\in E} c_e(x_{ij}+x_{ji}) + \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$\text{(6.14)}\quad \text{s.t.}\quad x_{ij}+x_{ji}\geq 1 \qquad \forall e=(i,j)\in E,$$

$$\text{(6.15)}\quad x(\delta^+(S))\geq 1 \qquad \forall S=\cup_{k\in Q}V_k,\ Q\subset\{1,\ldots,p\},$$

$$\text{(6.16)}\quad x(\delta^+(i))-x(\delta^-(i))+\sum_{(i,j)\in\delta(i)}(x_{ij}-x_{ji})=0 \qquad \forall i\in V,$$

$$\text{(6.17)}\quad x_{ij}\geq 0 \text{ and integer} \qquad \forall (i,j)\in A\setminus A_R,$$

$$\text{(6.18)}\quad x_{ij}\geq 1 \text{ and integer} \qquad \forall (i,j)\in A_R,$$

$$\text{(6.19)}\quad x_{ij},x_{ji}\geq 0 \text{ and integer} \qquad \forall (i,j)\in E.$$

Inequalities (6.14), called traversing inequalities, and (6.18) imply that all the required edges and arcs, respectively, are traversed. Connectivity inequalities (6.15) ensure that the solution is connected, while symmetry equations (6.16) force it to be symmetric.

6.2.1.4 ▪ MRPP polyhedron associated with formulation F2

Let $\text{MRPP}_{F2}(G)$ be the convex hull of all the vectors in $\mathbb{R}^{2|E|+|A|}$ satisfying (6.14) to (6.19). Although this polyhedron has not been studied directly, it can be seen as a particular case of the WGRP and WRPP polyhedra, studied in Plana [50] and Corberán, Plana, and Sanchis [25, 26]. In this section several results regarding the polyhedra of the WGRP and the WRPP are adapted to the MRPP.

The dimension of $\text{MRPP}_{F2}(G)$ is $2|E|+|A|-|V|+1$, and the following inequalities from the formulation are, under mild conditions, facet-defining for $\text{MRPP}_{F2}(G)$:

- Inequalities (6.14) and (6.15).
- Trivial inequalities $x_{ij}\geq 0\ \forall (i,j)\in A\setminus A_R$, $x_{ij}\geq 1\ \forall (i,j)\in A_R$, and $x_{ij},x_{ji}\geq 0$ $\forall (i,j)\in E$.

If integrality conditions are satisfied, symmetry inequalities (6.16) imply that all the vertices are even. However, once the integrality conditions have been removed from F2, symmetry conditions alone no longer guarantee that the vertices are of even degree. However, given an R-odd cutset $\delta^*(S)$, the following R-odd cut constraints must also be satisfied by any solution to F2:

$$\text{(6.20)}\quad x(\delta^*(S))\geq |\delta_R^*(S)|+1\ \forall \delta^*(S)\ R\text{-odd cutset.}$$

These R-odd constraints are also facet-inducing for $\text{MRPP}_{F2}(G)$ if graphs $G(S)$ and $G(V\setminus S)$ are strongly connected and $|\delta(S)|>\left||\delta_R^+(S)|-|\delta_R^-(S)|\right|$.

Furthermore, any valid inequality for the WRPP is also a valid inequality for the MRPP. However, the facet-inducing inequalities for WRPP(G) described in Section 6.3 are also facet-inducing for $\text{MRPP}_{F2}(G)$ only if the arcs in G are of the "appropriate direction," i.e., if the removed variables do not avoid the existence of all the tours needed to define a facet. The following inequalities, which can be easily adapted from the WRPP, are valid and facet-defining for $\text{MRPP}_{F2}(G)$ under some conditions (see Corberán, Plana, and Sanchis [26]):

- K-C and K-C_{02} inequalities.
- Path-Bridge and Path-Bridge_{02} inequalities.

- Honeycomb and Honeycomb$_{02}$ inequalities.
- Even and Odd zigzag inequalities.
- Even-even and Odd-odd zigzag inequalities.

6.2.1.5 ▪ Formulations comparison

Formulations F1 and F2 are compared in Corberán, Mota, and Sanchis [22] (see also Chapter 4 in this book). It is easy to see that both (integer) formulations are equivalent. However, the lower bounds obtained from their LP-relaxations can be different.

Consider the following initial LP relaxation, LP0$_{F1}$, corresponding to formulation F1, containing

- constraints $x_e \geq 1 \ \forall e \in E \cup A_R$ and $x_e \geq 0 \ \forall e \in A \setminus A_R$,
- one connectivity inequality (6.3) for each R-set,
- one balanced-set inequality (6.4) for each "unbalanced" vertex,
- the system equations (6.7), and
- one R-odd cut inequality (6.8) for each R-odd "balanced" vertex,

and LP0$_{F2}$, associated with formulation F2, as the LP containing

- constraints $x_{ij} \geq 0 \ \forall (i,j) \in A \setminus A_R \ x_{ij} \geq 1 \ \forall (i,j) \in A_R$, and $x_{ij}, x_{ji} \geq 0 \ \forall (i,j) \in E$,
- constraints (6.14),
- one connectivity inequality (6.15) for each R-set,
- one symmetry equation (6.16) for each vertex, and
- one R-odd cut inequality (6.20) for each R-odd "balanced" vertex.

It is shown in Corberán, Mota, and Sanchis [22] that LP0$_{F2}$ is stronger than LP0$_{F1}$. Now consider the above formulations after adding all the families of inequalities that are known to be separable in polynomial time, which are connectivity, balanced-set, and R-odd cut inequalities. In [22] it is also shown that both extended formulations produce the same bound, while the computational effort to solve them is, at least in theory, comparable.

6.2.2 ▪ Exact methods

In Blais and Laporte [12] it is shown how the MRPP can be solved exactly by transforming it into an equivalent vertex routing problem for which an exact solution procedure is available. To transform an MRPP on a mixed graph $G = (V, E, A)$ into an equivalent vertex routing problem on a directed graph $\widetilde{G} = (\widetilde{V}, \widetilde{A})$, each required arc $(i,j) \in A_R$ is replaced by a required vertex v_{ij}, and each required edge $e = (i,j) \in E_R$ is first replaced by two opposite arcs (i,j) and (j,i) of cost $\widetilde{c}_{ij} = \widetilde{c}_{ji} = c_e$ and then substituted for with two vertices v_{ij} and v_{ji}, only one of which is required. The vertex set $\widetilde{V}$ is then made up of all the vertices just defined. Each vertex pair (v_{ki}, v_{lj}) in the transformed problem defines an arc of $\widetilde{A}$ having a cost $\widetilde{c}_{ij} = s_{il} + c_{lj}$, where s_{il} represents the cost of the shortest path in G from vertex i to vertex l and $\widetilde{c}_{ii} = 0$. This transformation is depicted in Figure 6.7, taken from Blais and Laporte [12].

If $E_R \neq \emptyset$, the resulting problem on $\widetilde{G}$ is not a TSP, but a *Generalized TSP* (GTSP), since only one of the two vertices v_{ij} and v_{ji} corresponding to a required edge $e = (i,j)$

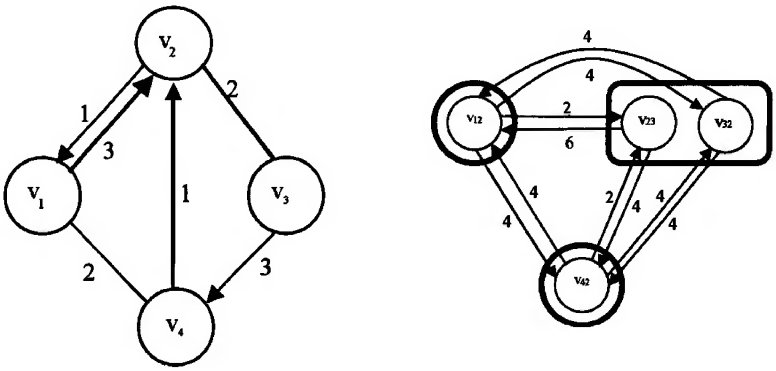


Figure 6.7. *Graph transformation for a mixed graph [12].*

must appear in the solution. To obtain the appropriate GTSP, the vertex set $\widetilde{V}$ is partitioned into clusters, with a cluster $\{v_{ij}, v_{ji}\}$ for each required edge $e = (i, j)$. The GTSP then consists of determining a minimum cost Hamiltonian circuit on $\widetilde{G}$ passing through each cluster exactly once. Solving this asymmetric GTSP on $\widetilde{G}$ provides a solution for the original MRPP on G. The asymmetric GTSP is solved using two methods. The direct approach applies the Lagrangian-based branch-and-bound algorithm proposed in Noon and Bean [47]. The undirected one first transforms the asymmetric GTSP into an equivalent *Asymmetric TSP* (ATSP) and then solves it by means of the exact method described in Carpaneto, Dell'Amico, and Toth [13]. The transformation procedure was assessed on randomly generated mixed graphs with $|V| = 500$, $|E| = |A| = 1000$, and with up to $|A_R| = 200$ and $|E_R| = 10$ or $|A_R| = 10$ and $|E_R| = 25$. The authors point out that the difficulty of the problem is clesely related to the size of $|E_R|$, which determines the number of clusters in the GTSP. These exact procedures were able to solve instances involving up to 25 required edges in G and up to 230 vertices in $\widetilde{G}$.

Corberán, Romero, and Sanchis [27] implement a preliminary cutting-plane algorithm for the MRPP using formulation F1 and including separation procedures only for connectivity, R-odd cut, and balanced-set inequalities. Later on, Corberán, Mejía, and Sanchis [21] improve it by adding separation procedures for the K-C, Honeycomb (with $L=1$), and regular Path-Bridge inequalities. The initial LP relaxation used is the one denoted above as LP0_{F1}. As mentioned before, the separation of connectivity, R-odd cut, and balanced-set inequalities is done in polynomial time, while the authors apply heuristic algorithms for the separation of K-C, regular Path-Bridge, and Honeycomb inequalities. When the separation procedures do not find violated inequalities and the current LP solution x^* is not a tour for the MRPP, a branch-and-bound procedure is applied. If the resulting integer solution is a tour, it is optimal. Otherwise, the procedure terminates with a tight lower bound, but no feasible solution. This algorithm was tested on 100 randomly generated MRPP instances with the following features: $3 \leq p \leq 12$, $20 \leq |V| \leq 100$, $15 \leq |E| \leq 220$, $5 \leq |E_R| \leq 150$, $50 \leq |A| \leq 350$, $5 \leq |A_R| \leq 200$. All of these instances were solved to optimality.

In Corberán, Plana, and Sanchis [25], a branch-and-cut algorithm for the WRPP is presented. This algorithm is based on a formulation with two variables per edge similar to formulation F2 for the MRPP and will be described in detail in Section 6.3.2. It was tested on 117 MRPP instances with up to 1000 vertices, 2000 edges, and 2000 arcs, solving all of them to optimality within an average time of half an hour.

6.2.3 ▪ Heuristic methods

A constructive heuristic and a tabu search algorithm for the MRPP are described in Corberán, Martí, and Romero [20]. The heuristic algorithm consists of four stages. In the first stage, a subset of arcs is chosen so that the graph including the required links of G and this subset of arcs is connected. In the second stage, the connectivity of this new graph is improved by solving a minimum cost flow problem in a directed graph, as in the heuristic algorithm proposed in Christofides et al. [15]. A feasible solution is obtained in the third stage. First, another minimum cost flow is computed in order to construct a balanced mixed graph, which is then made even by solving a minimum cost matching on the odd-degree vertices, and the added edges are oriented to get a symmetric directed graph. Finally, stage four tries to improve the MRPP solution.

The tabu search procedure starts from an MRPP tour obtained with the heuristic algorithm described above. In order to move from one solution to another one, the tabu search procedure uses the following movement: Let (i, j) be an arc connecting two R-sets in the current solution, and let $[k, i]$ and $[j, s]$ be paths from k to i and from j to s, respectively, in the augmentation of the graph induced by the required links. Then the movement consists of deleting path $[[k, i], (i, j), [j, s]]$ from the current solution and adding the shortest path from k to s. Note that the resulting graph may be disconnected, and thus it would not be a feasible MRPP solution. The tabu search algorithm is divided into two phases: intensification and diversification. At each iteration of the intensification phase, an arc is chosen with a probability proportional to its cost and the corresponding movement described above is applied. The removed arc is marked as tabu so that it is not added again in the intensification phase. This phase ends when no arc is found for which a movement can be applied. The diversification phase is applied for a fixed number of iterations. At each iteration a non-tabu arc connecting R-sets is chosen at random as in the intensification phase. If the corresponding movement can be applied, it is performed; otherwise, two alternative movements are considered. If the current solution is connected, the shortest path to be added in the movement is computed in the graph obtained by deleting the tabu arcs. If the current solution is nonconnected, a connected component is randomly selected and all the arcs connecting this component for which the associated shortest path does not include a tabu arc are considered. The lowest cost one, considering the arc plus the shortest return path costs, is selected. If all the above paths include a tabu arc, we ignore the tabu status. Both phases are alternated until the best-known solution for a fixed number of consecutive iterations is not improved. Additionally, a long-term diversification phase is implemented to complement the diversification phase in the basic procedure. This phase consists of a restarting mechanism in which information generated by the search history is used to guide the construction process.

Both the constructive heuristic and the tabu search algorithm were tested on a set of 100 randomly generated instances. These instances have up to 12 R-sets, 100 vertices, 220 edges, 150 required edges, 350 arcs, and 200 required arcs. The constructive method presented an average deviation of 1.87% with respect to a lower bound obtained by the cutting-plane algorithm described in Corberán, Romero, and Sanchis [27]. In three of the 100 instances, the solution provided matched the lower bound and was thus proved to be optimal. The tabu search procedure was tested using different sets of parameters. For all the combinations tried, the average deviation was less than or equal to 0.53%, with the best one obtaining an average deviation of 0.48%. The average time used by the different versions of the tabu search ranges from 57.8 to 326.3 seconds on a Pentium-150 PC, while the time consumed by the constructive algorithm was negligible.

Another heuristic procedure for the MRPP is the one proposed by Chyu [17]. The

author applies this algorithm to the special case in which for each arc (i,j) in the graph there is an opposite arc (j,i) and both arcs have the same cost. The heuristic includes four subheuristics, with each one being designed for some particular structure of the required arcs set. The best result produced among the four subheuristics is then selected. This heuristic has a worst-case performance ratio of between 3/2 and 15/8, depending on the ratio of the cost of the required edges to that of the required arcs in the problem.

6.3 ▪ The windy Rural Postman Problem

In this section we define the Windy Rural Postman Problem (WRPP) and describe its heuristic and exact solution. Consider a windy graph $G=(V,E)$ with a vertex set V and an edge set E, and two nonnegative costs c_{ij}, c_{ji} for each $(i,j)\in E$, representing the costs of traversing it in each direction. Given a subset of required edges $E_R \subseteq E$, the WRPP consists of finding a minimum cost tour traversing all the required edges at least once.

6.3.1 ▪ Problem formulation

The formulation for the WRPP was first proposed in Benavent et al. [8]. In order to present it, we need some notation. Given $S \subseteq V$, $\delta(S)$ denotes the set of edges with an endpoint in S and the other in $V \setminus S$. Given $S_1, S_2 \subseteq V$, $(S_1 : S_2)$ represents the set of edges with one endpoint in S_1 and the other in S_2, while $\delta_R(S)$ and $(S_1 : S_2)_R$ denote the previous sets referring only to the required edges. A vertex is R-even (R-odd) if it is incident with an even (odd) number of required edges, and a subset of vertices $S \subseteq V$ is called R-even (R-odd) if it contains an even (odd) number of R-odd vertices. As in the MRPP, the connected components of the graph induced by the required edges, $G_R = (V,E_R)$, denoted by $V_1,\ldots,V_p$, are called R-sets.

The formulation uses two variables x_{ij} and x_{ji} associated with each edge (i,j), representing the number of times edge (i,j) is traversed from i to j and from j to i, respectively. Given $F \subseteq E$, $x(F)$ denotes the sum of all the variables associated with the edges in F, $x(F)=\sum_{(i,j)\in F}(x_{ij}+x_{ji})$, and given $S_1, S_2 \subset V$, $x(S_1,S_2)$ denotes the sum of the variables associated with the traversal of the edges in $(S_1 : S_2)$ from S_1 to S_2, that is $x(S_1 : S_2) = x(S_1,S_2)+x(S_2,S_1)$. The WRPP formulation is

$$\text{(6.21)} \qquad \text{(WRPP)} \quad \min \sum_{(i,j)\in E} (c_{ij}x_{ij} + c_{ji}x_{ji})$$

$$\text{(6.22)} \qquad \text{s.t.} \quad x_{ij} + x_{ji} \geq 1 \qquad \forall (i,j)\in E_R,$$

$$\text{(6.23)} \qquad x(S, V\setminus S) \geq 1 \qquad \forall S = \cup_{k\in Q} V_k,\ Q \subset \{1,\ldots,p\},$$

$$\text{(6.24)} \qquad \sum_{(i,j)\in\delta(i)} (x_{ij} - x_{ji}) = 0 \qquad \forall i \in V,$$

$$\text{(6.25)} \qquad x_{ij}, x_{ji} \geq 0 \qquad \forall (i,j)\in E,$$

$$\text{(6.26)} \qquad x_{ij}, x_{ji} \text{ integer} \qquad \forall (i,j)\in E.$$

Inequalities (6.22) imply that each required edge is traversed. Connectivity inequalities (6.23) and symmetry equations (6.24) ensure that the tours are connected and symmetric, respectively.

6.3.1.1 ▪ The WRPP polyhedron

Let WRPP(G) denote the convex hull of all the tours for the WRPP on graph G, i.e., all the integer vectors $x \in \mathbb{R}^{2|E|}$ satisfying (6.22) to (6.25). In Plana [50] and Corberán, Plana, and Sanchis [26] it is shown that WRPP(G) is an unbounded polyhedron of dimension $2|E|-|V|+1$ and that inequalities (6.22), (6.23), and (6.25) are facet-inducing for WRPP(G) under mild conditions.

In Benavent et al. [8], it is also shown how to obtain valid inequalities for an asymmetric Arc Routing Problem from inequalities for its symmetric version. Let $\mathscr{P}$ be an Arc Routing Problem defined on an undirected graph $G = (V,E)$ and assume that it is formulated with a variable y_e associated with each edge $e = (i,j) \in E$, representing the number of times the edge is traversed in any direction. Let $\mathscr{P}'$ be the "windy" version of this Arc Routing Problem, i.e., the problem defined on the same graph $G = (V,E)$, but with two nonnegative costs associated with each edge. Assume that this problem is formulated with two variables x_{ij} and x_{ji} associated with each edge $(i,j) \in E$, representing the number of times the edge is traversed in each direction. Then the following result holds:

Let $\sum_{e\in E} \alpha_e y_e \geq \alpha_0$ be a valid inequality for $\mathscr{P}$. The corresponding inequality

$$\sum_{e=(i,j)\in E} \alpha_e (x_{ij} + x_{ji}) \geq \alpha_0$$

is valid for $\mathscr{P}'$ if, for every feasible solution $x \in \mathbb{Z}^{2|E|}$ for problem $\mathscr{P}'$, the vector $y \in \mathbb{Z}^{|E|}$, defined as $y_e = x_{ij} + x_{ji}$ for all $e = (i,j) \in E$, is a feasible solution for problem $\mathscr{P}$.

Since problems defined on directed or mixed graphs can be considered as windy problems where some variables have been set to zero, the above result implies that

- All the valid inequalities for the (Undirected) Chinese Postman Problem produce valid inequalities for the Directed and Mixed Chinese Postman Problems and for the Windy Postman Problem, and
- From the (Undirected) Rural Postman Problem valid inequalities for their Directed, Mixed, and Windy versions can be obtained.

Moreover, the authors conjecture that, if the condition of the above result holds, the following result is also true:

Given a facet-inducing inequality $\sum_{e\in E} \alpha_e y_e \geq \alpha_0$ for the polyhedron associated with problem $\mathscr{P}$, the corresponding inequality $\sum_{e=(i,j)\in E} \alpha_e (x_{ij} + x_{ji}) \geq \alpha_0$ is facet-inducing for the polyhedron associated with problem $\mathscr{P}'$.

Although, as far as we know, this conjecture has not been proved yet, the following families of inequalities, some of them derived from the undirected RPP, are known to be facet-inducing for WRPP(G).

R-odd cut inequalities

Given that any tour must cross any given edge cutset an even number of times, the following R-odd cut inequalities:

$$x(\delta(S)) \geq |\delta_R(S)| + 1 \quad \forall S \subset V \quad \text{such that } \delta(S) \text{ is } R\text{-odd}, \tag{6.27}$$

are valid for WRPP(G). They are facet-inducing for WRPP(G) if subgraphs $G(S)$ and $G(V \setminus S)$ are connected (Corberán, Plana, and Sanchis [26]).

K-C and K-C_{02} inequalities

K-C and K-C_{02} inequalities have already been described for the MRPP in this chapter. For the WRPP, these inequalities are based on the same partition of V and have the same right-hand side and the same coefficients of the corresponding inequalities for the MRPP (see Figure 6.4). In particular, given a variable associated with the traversal of an edge in a given direction, its coefficient is the same as the corresponding arc in a K-C or K-C_{02} inequality for the MRPP. Both K-C and K-C_{02} inequalities are valid and facet-inducing for WRPP(G) (Corberán, Plana, and Sanchis [26]).

Path-Bridge, Path-Bridge$_{02}$, Honeycomb, and Honeycomb$_{02}$ inequalities

Again, the partition of V in which these inequalities are based, the coefficients of the variables, and their right-hand sides are the same as those of the corresponding inequalities for the MRPP (see Section 6.2.1.2). All of them are valid and facet-inducing for WRPP(G) (Corberán, Plana, and Sanchis [26]).

Zigzag inequalities

When solving the WRPP, it is not unusual to find fractional solutions containing closed walks resembling a zigzag whose edges satisfy $x_{ij} + x_{ji} = 1.5$ if $(i,j) \in E_R$ and $x_{ij} + x_{ji} = 0.5$ otherwise. Consider, for example, the vector represented in Figure 6.8(a), where arcs drawn in solid lines correspond to variables of value 1, while arcs in dotted lines correspond to variables of value 0.5. This vector satisfies all the previously described inequalities but can be cut by means of zigzag inequalities. There are four families of zigzag inequalities. The first two, Odd and Even zigzag inequalities, were introduced in Corberán, Plana, and Sanchis [24], while the other two, Odd-odd and Even-even zigzag inequalities, were proposed in Corberán et al. [23] for the *Windy Postman Problem* (WPP) and extended to the WRPP in Corberán, Plana, and Sanchis [26].

Consider a partition of set V into four subsets, M^1, M^2, M^3, and M^4. Let α_{ij} denote the number of required edges in $(M^i : M^j)$, and suppose one of the following conditions is satisfied:

Even case: M^1, M^2, M^3, and M^4 are R-even, and $\alpha_{12} = \alpha_{34} = \alpha_{14} = \alpha_{23} = 0$.

Odd (simple) case: M^1, M^2, M^3, and M^4 are R-odd, and $\alpha_{12} + \alpha_{34} = \alpha_{14} + \alpha_{23}$.

The Even and Odd (simple) zigzag inequalities are

$$x(\delta(M^1 \cup M^2)) + 2x(M^2,M^1) + 2x(M^4,M^3) \geq \alpha_{13} + \alpha_{24} + \alpha_{14} + \alpha_{23} + 2. \tag{6.28}$$

Examples of these inequalities are represented in Figures 6.8(b) and 6.8(c), where the required edges are represented in bold lines and each number represents the coefficient of the variable associated with the traversal of the edge from the nearest vertex to the farthest one.

In Corberán, Plana, and Sanchis [24] it is shown that inequalities (6.28) are valid and facet-inducing for WRPP(G) if the following conditions hold: if $(M^i : M^j) \neq \emptyset$, $\forall i,j$, and $\alpha_{13}, \alpha_{24} \geq 2$ (even case) or $\alpha_{13} \geq |\alpha_{12} - \alpha_{14}| + 1$ and $\alpha_{24} \geq |\alpha_{12} - \alpha_{23}| + 1$ (odd case). More

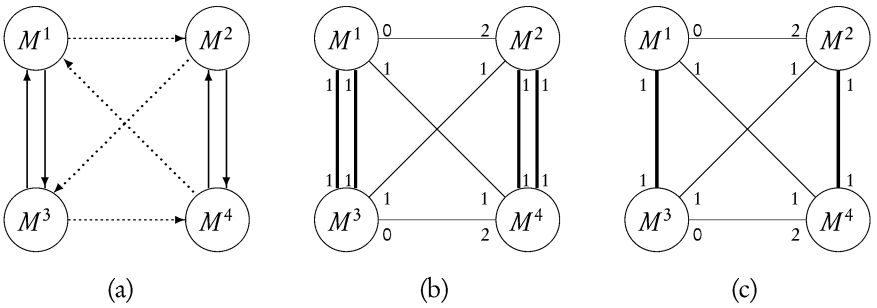


Figure 6.8. *Fractional solution and Even and Odd zigzag inequalities.*

general Odd zigzag inequalities are also proposed in the same paper (see also Chapter 4 in this book).

The two other families of inequalities, the Even-even and the Odd-odd zigzag inequalities, are defined by a partition of the set of vertices into four subsets, just as with the Odd and Even zigzag inequalities. However, in this case not all of these subsets have to be incident with an odd, respectively even, number of required edges. As mentioned above, they were introduced for the WPP in Corberán et al. [23] (see also Chapter 4) and extended to the WRPP in Corberán, Plana, and Sanchis [26], where it is proved that they are valid and facet-inducing under certain conditions.

6.3.2 ▪ Exact methods

In Corberán, Plana, and Sanchis [25], a branch-and-cut algorithm to solve the WRPP is presented. This procedure starts with an initial LP including constraints (6.22), (6.24), and (6.25), as well as one connectivity constraint (6.23) associated with each R-set. Furthermore, R-odd cutset inequalities (6.27) associated with R-odd vertices and with the connected components S of the subgraph of G_R induced by the R-odd vertices, if $|S|$ is odd, are also included.

At each iteration of the cutting-plane algorithm, exact and heuristic separation algorithms are used for identifying violated R-odd cut and connectivity inequalities. Moreover, heuristic algorithms for separating K-C, K-C_{02}, HC, HC_{02}, PB, Even, and Odd zigzag inequalities are used. The cutting-plane procedure is applied at each node of the tree until no new violated inequalities are found or until a stopping criterion, called tailing-off, is satisfied.

The constructive algorithms H1 and H2 described in Section 6.3.3 are used to obtain an initial upper bound. If the gap at the root node with respect to this upper bound is greater than 1.5%, an artificial upper bound with a value 1.005 times the lower bound is used. If the procedure ends without finding an optimal solution, it is restarted using a greater artificial upper bound (1.010 times the lower bound). If, again, the algorithm ends unsuccessfully, it is restarted for the last time using the true upper bound.

This branch-and-cut algorithm was tested on 522 WRPP instances with up to 1000 vertices, 4000 edges, and 200 R-sets, obtaining 510 optimal solutions. The time limit was set to 10 hours, but the average time used by the algorithm for solving the hardest instances (those with around 1000 vertices) to optimality was less than 1500 seconds on a Pentium IV at 2.8 GHz machine with 1GB RAM.

Another exact procedure for the WRPP is described in Afsar, Jozefowiez, and Lopez

[1]. It is a branch-and-price algorithm that has been tested on 198 of the smallest instances used in Corberán, Plana, and Sanchis [25], with up to 100 vertices and 184 edges. All of the instances but three were solved to optimality. For the solved instances, this procedure took less than five seconds, except one instance which took a little longer than 200 seconds on an Intel dual core 3.0 GHz machine with 3GB RAM.

6.3.3 ▪ Heuristic methods

Here we briefly describe several constructive and metaheuristic algorithms for the WRPP presented in Benavent et al. [8, 9].

6.3.3.1 ▪ Constructive algorithms

In Benavent et al. [8], three heuristic algorithms for the solution of the WRPP are presented. The first algorithm, WRPP1, consists of three phases. First, the components of graph G_R are joined to obtain a connected graph by adding the edges corresponding to a shortest spanning tree. Then, a minimum cost matching problem on the odd-degree vertices of the resulting graph is solved, obtaining a connected and even windy graph. Win's exact algorithm for the WPP defined on Eulerian graphs (Win [55]), described in Chapter 4, is then applied to obtain a feasible solution for the WRPP.

The second heuristic, WRPP2, begins by assigning the direction associated with its minimal traversing cost to each edge in G_R, in order to obtain a directed graph G_R^d. Then a balanced mixed graph G^1 is obtained by solving a minimum cost flow on a special graph. If G^1 has odd-degree vertices, a minimum cost matching problem is solved to obtain an even mixed graph G^2. Again, the edges in G^2 are oriented in the direction of their minimal traversing cost and a new network flow problem is solved to obtain a symmetric digraph G^3. Finally, as graph G^3 may be disconnected, a last phase consisting of the solution of a shortest spanning tree connecting its components is executed.

The third constructive heuristic, WRPP3, is simpler and consists of first computing a shortest spanning tree connecting the components of G_R, then orienting the edges according to the minimum traversal cost, and finally solving a transportation problem to get a connected and symmetric directed graph that defines a feasible WRPP solution.

The solutions generated by all the algorithms are improved by means of two simple procedures described in the same paper. The first one removes redundant cycles from the solution graph, while the second one consists of first constructing an Eulerian tour traversing the solution graph and then substituting any path of nonrequired arcs connecting two consecutive required edges with a shortest path.

The above algorithms have no finite worst-case bound (Benavent et al. [8]). In fact, contrary to what happens with the WPP, it is not known if the WRPP admits a polynomial time heuristic with a constant worst-case bound. As far as we know, this is an open question.

In Benavent et al. [9], two new constructive heuristics, several multistart algorithms, and a Scatter Search procedure for the WRPP are presented. The constructive heuristics, called H1 and H2, are obtained by modifying the previous heuristics WRPP1 and WRPP2, respectively, in a similar way Raghavachari and Veerasamy [51] do to the Frederickson heuristic for the Mixed Chinese Postman Problem (Frederickson [34]). Local search based on 2- and Or-interchanges (Or [48]) is applied, as well as another method, based on the one proposed in Lacomme, Prins, and Ramdane-Chérif [43]. Given an order of traversal of the required edges, this method, called the reversing direction procedure, consists of finding the optimal direction in which each required edge has to be traversed.

The multistart algorithms are based on introducing some random elements into the constructive heuristics described above. The one producing the best results uses the heuristic WRPP2 with randomized costs in the computation of the matching and the shortest spanning tree. The improvement phase consists of applying first the Or-interchange, then the 2-interchange, and finally the reversing direction procedure.

The Scatter Search procedure starts with the construction of an initial reference set of solutions (RefSet) from a large set P of diverse solutions. This initial population P contains the five solutions provided by the constructive algorithms described earlier and is completed with solutions generated with the construction phase of the best multistart algorithm. The k best solutions in P are included in RefSet. Another l solutions are added in such a way that the diversity of RefSet is as large as possible. To combine the solutions in the reference set, a version of the classical order crossover similar to that used in Lacomme, Prins, and Ramdane-Chérif [42] is applied. A local search phase consisting of the Or-interchange, the 2-interchange, and the reversing direction procedures is applied to the resulting solution. The procedure stops when, after combining all the solutions in RefSet, no new solutions are incorporated.

Table 6.1 (taken from Benavent et al. [9]) summarizes the results obtained with the best constructive algorithm (H2), the best multistart (MS), and the Scatter Search (SS) on four sets of randomly generated instances. Their sizes and characteristics are similar to those of the 117 instances mentioned at the end of the previous section. For each set of instances, the average gap, number of optimal solutions found, and average time in seconds is reported. The gap is computed with respect to a lower bound obtained with the cutting-plane procedure described in Benavent et al. [8].

Table 6.1. *Results obtained with heuristic algorithms for the WRPP.*

		H2	MS	SS
Albaida	Average gap (%)	2.21	0.45	0.29
(72 instances)	# of optima	8	22	21
	Time (s)	0.11	150.46	4.06
Madrigueras	Average gap (%)	2.23	0.61	0.44
(72 instances)	# of optima	5	14	12
	Time (s)	0.43	162.53	11.86
500 vertices	Average gap (%)	1.38	1.38	1.06
(27 instances)	# of optima	0	0	0
	Time (s)	2.95	200	145.96
1000 vertices	Average gap (%)	0.72	0.72	0.63
(27 instances)	# of optima	1	1	1
	Time (s)	26.88	200	1543.606

6.4 ▪ Related problems

In this section we present the results obtained for some problems closely related to the MRPP and WRPP. Among them, the DRPP is addressed, as well as the *Stacker Crane Problem* (SCP), which, although it can be considered a special case of the MRPP and DRPP, has some special features that recommend its separate presentation. Other variants described here are more related to real-life applications, like those with turn penalties and

forbidden tours or zigzag servicing, which allows the servicing of the two sides of a street segment simultaneously.

6.4.1 ▪ The directed Rural Postman Problem

As with the MRPP, the DRPP can be solved exactly by transforming it into an ATSP (Blais and Laporte [12]). To transform a DRPP on a directed graph $G = (V,A)$ into an equivalent vertex routing problem on a directed graph $\widetilde{G} = (\widetilde{V},\widetilde{A})$, each required arc $(i,j) \in A_R$ is replaced by a required vertex v_{ij}. These vertices define the vertex set $\widetilde{V}$. Associated with each pair of vertices (v_{ki}, v_{lj}) there is an arc of $\widetilde{A}$ with cost $\widetilde{c}_{ij} = s_{il} + c_{lj}$, where s_{il} represents the cost of the shortest path in G from vertex i to vertex l, and c_{lj} is the cost of the arc (l,j) in G. The resulting problem on graph $\widetilde{G}$ is an ATSP, whose solution solves the original DRPP on G. The exact method used in Blais and Laporte [12] for solving the ATSP is the one described in Carpaneto, Dell'Amico, and Toth [13]. With this methodology, the authors were able to solve randomly generated directed graphs with $|V| = 5000$ and $|A| = 50000$, and with up to $|A_R| = 3500$.

A direct algorithm for the solution of the *Directed General Routing Problem* (DGRP), and the DRPP as special case, is proposed in Ávila et al. [5] and Ávila [3]. A formulation of the problem, similar to formulation MRPP-F2 presented in Section 6.2.1.3, is used to define the DGRP polyhedron. Several families of inequalities are proved to be facet-inducing of this polyhedron: trivial, connectivity, K-C, Path-Bridge, and a new family, the Asymmetric 2PB inequalities. Based on this polyhedral study, the authors implement a branch-and-cut procedure that is capable of solving to optimality 36 randomly generated DRPP instances with 1000 vertices and up to 3200 arcs and 553 R-sets. Moreover, 70 instances generated with the same characteristics as those used in Blais and Laporte [12] have been optimally solved in short computing times.

6.4.2 ▪ The stacker crane problem

The SCP basically consists of finding a tour that starts and ends at a given vertex and traverses the set of arcs (representing jobs) of a mixed graph at minimum cost. Its name refers to the practical problem of operating a crane. The crane must start from an initial position, perform a set of movements, and return to the initial position. The objective is to schedule the movements of the crane so as to minimize the total tour cost. This problem can be considered a special case of both the MRPP and the DRPP. The SCP was proposed by Frederickson, Hecht, and Kim [35]. They defined the SCP as an arc routing problem on a mixed graph where a given vertex represents the depot, i.e., the initial and final position of the crane. Each arc and edge of the graph has an associated nonnegative cost. The goal is to find a minimum cost tour, starting and finishing at the depot, which traverses all the arcs in the graph.

It is shown in Frederickson, Hecht, and Kim [35] that the SCP is NP-hard, and a heuristic algorithm with a worst-case performance ratio of 9/5 and $O(n^3)$ complexity is developed. This procedure consists of two algorithms: The first one computes a minimum-cost matching and then a minimum-cost spanning tree. The second one transforms the SCP instance into a TSP one and then uses Christofides' algorithm for the TSP [14]. Both procedures need the costs of the instances to satisfy the triangle inequality. The first procedure works better if the cost of the arcs is large compared to the cost of the edges of an optimal solution, while the second one works better in the opposite situation.

In Berbeglia et al. [11], the SCP is presented as a pickup and delivery problem. This is

an important class of vehicle routing problems in which commodities or people have to be transported from origins to destinations. They have been the object of study in recent years because of their many applications in logistics, ambulatory services, and robotics. The SCP can be considered as a pick-up and delivery problem where single objects have to be transported from their origins to their destinations using a unit capacity vehicle.

Other papers dealing with the SCP can be found in the literature. Eiselt, Gendreau, and Laporte [33] present a survey on the RPP and devote a section to the SCP. Zhang [56] proposes a simplification of Frederickson's algorithm with a worst-case ratio of 2 and $O(n^2)$ complexity. Zhang and Zheng [57] transform the SCP into an ATSP and solve it using existing algorithms for the ATSP. Hassin and Khuller [38] propose a $\frac{1}{2}$ z-approximation algorithm (a polynomial-time algorithm that produces a solution whose distance from the optimal one is at most $\frac{1}{2}$ times the distance between the optimal solution and the worst possible solution) for the ATSP that can also be used for the SCP. Srour and van de Velde [54] study the difficulty of the SCP, presenting a statistical study that compares the difficulty of the solution of SCP instances with that of general ATSP instances. The motivation for this study is the claim by some authors that although the SCP and the ATSP are known to be NP-hard, the SCP can be easily solved when transformed into an ATSP (Laporte [44]). Their conclusion is that SCP instances are not necessarily easier than other ATSP instances, but a special subset of SCP instances, called drayage problems, are more readily solved. The study by Cirasella et al. [18] of ATSP heuristics on 11 types of instances contains one type corresponding to the SCP.

In Ávila et al. [4], a metaheuristic algorithm based on the combination of a Multistart and an Iterated Local Search algorithm is presented. This algorithm was tested on several SCP instances generated from 92 instances for the RPP proposed in Hertz, Laporte, and Nanchen-Hugo [39], providing good solutions in reasonable computing times.

More recently, the SCP has been studied as a special case of the DGRP in Ávila et al. [5] and Ávila [3]. The branch-and-cut algorithm proposed in these works has been tested on instances randomly generated on grids of size 50×50 and 100×100 with a number of jobs varying from 100 to 3000. All the instances defined on a 50×50 grid have been solved to optimality in very short computing times. Also, 27 out of 40 instances generated on a 100×100 grid have been solved to optimality within the time limit of one hour. For those instances that could not be solved optimally, the average gap between the final lower bound and the best solution found was less than 0.33%.

6.4.3 ▪ The DRPP and MRPP with turn penalties

When a real-life problem is modeled and solved as an MRPP, or as a particular case of it, it is commonly assumed that all turns are allowed and that they are not time consuming. While some turns can be considered irrelevant, others are more time consuming and/or dangerous in their execution, and most U-turns may even be forbidden. Therefore, the MRPP solution could be unfeasible if the traffic signals are respected, especially inside a city.

To model these situations, Benavent and Soler [10] and Corberán et al. [19] introduce, respectively, the DRPP and MRPP with Turn Penalties, which take into account turn penalties and forbidden turns. Both papers provide theoretical results on the complexity of these problems and propose a transformation into an ATSP that is then solved using already known algorithms. In Soler, Martínez, and Micó [53], a more general problem is considered: the MGRP with Turn Penalties and forbidden turns. In this problem, the route must visit all the required vertices and traverse all the required arcs and edges of the graph, without making forbidden turns. Again, the authors propose a transformation

of this problem into an ATSP. In this paper, however, as the ATSP can be considered a particular case of the MGRP, the authors use the exact algorithm for the MGRP described in Corberán, Plana, and Sanchis [25] instead of the known ATSP algorithms. Computational results on a set of 128 instances with up to 200 vertices, 440 arcs, 176 required arcs, 26 required vertices, and 40 required edges are presented. 120 of these 128 instances were solved to optimality in less than two hours of computing time.

6.4.4 ▪ The directed clustered RPP

The *Directed Clustered Rural Postman Problem* (DCRPP) is a restricted version of the DRPP in which each connected component of required arcs has to be completely serviced before starting the service of another component. The use of arcs from different connected components for deadheading is allowed. This problem is introduced by Dror and Langevin [32]. The motivation for this problem stems from the observation that, in the case of postal delivery, for example, the mail is presorted to facilitate the speed of mail distribution and is usually assigned in connected components, forcing the completion of a traversal of a given subset of arcs before starting the traversal of a new subset. Other applications appear in the context of torch cutting and the servicing of parking meters. In Dror and Langevin [32], the DCRPP is first transformed into an Asymmetric Generalized TSP (AGTSP) and then solved exactly by a Lagrangian-based method by Noon and Bean [47]. The authors succeeded in solving random instances of up to 81 vertices, 247 arcs (including 88 required arcs), and 15 connected components.

6.4.5 ▪ The generalized DRPP

The *Generalized Directed Rural Postman Problem* (GDRPP), also known as the *Close-Enough Arc Routing Problem*, is an arc routing problem with some interesting real-life applications, such as routing for meter reading. Let $G = (V,A)$ be a directed graph with set of vertices V and set of arcs A, and let $c_{ij} \geq 0$ be the cost associated with the traversal of arc $(i,j) \in A$. Given a family of arc subsets $\mathbb{H} = \{H_1,\ldots,H_L\}$, the GDRPP consists of finding a minimum cost tour starting and ending at the depot and traversing at least one arc from each subset H_m, $m = 1,\ldots,L$. These subsets H_m do not need to be disjoint nor induce connected subgraphs.

This problem, without considering the presence of a depot, was introduced by Drexl [30]. He noted that the GDRPP is NP-hard since, when each set H_m consists of a single arc, the GDRPP reduces to the DRPP, and proposed a formulation and a branch-and-cut algorithm producing good computational results. A more recent version of his work was published in [31].

However, it was Shuttleworth et al. [52] who presented this problem in the context of constructing routes for meter reading, calling it the *Close-Enough Traveling Salesman Problem*. In this application, a vehicle with a receiver travels through a series of neighborhoods. If the vehicle gets closer than a certain distance to a meter (customer), the receiver is able to record the gas, water, or electricity consumption. Therefore, the vehicle does not need to traverse every street, but only a few, in order to get close enough to each meter. The set of streets from which a certain meter m can be read defines the set H_m. In their work, Shuttleworth et al. proposed four heuristics to solve eight real-life instances with an average of 900 customers and 9000 streets each. The procedures have essentially two phases: First a subset of streets to be traversed is selected, and then a route starting and ending at the depot and traversing all these streets is found. The first phase is obviously related to the resolution of a Set Covering Problem (SCP), while the second consists of

solving a DRPP (or a Directed General Routing Problem if the depot is not incident with a selected street).

More recently, Hà et al. [37] studied this problem, which they called the Close-Enough Arc Routing Problem, and proposed a new formulation, which they compared with a previous one introduced by the same authors [36] and the one by Drexl [30, 31]. Moreover, they proposed a branch-and-cut algorithm providing very good computational results on large-size instances.

In Ávila et al. [6] a slightly different formulation is presented and its associated polyhedron is studied. The authors describe several new families of valid inequalities that are used in a branch-and-cut algorithm, whose performance improves that of the branch-and-cut by Hà et al. [37].

6.4.6 ▪ The WRPP with zigzag service

The *WRPP with zigzag service* (WRPPZ) is a variant of the WRPP proposed by Irnich [40]. In this problem, the streets can be divided into four classes. First, street segments with houses on one side of the street only, which must be serviced exactly once. Second, street segments with houses on both sides, where the sides must be serviced separately and where, consequently, the segments must be traversed at least twice. Third, street segments with houses on both sides with the option of servicing both sides simultaneously by a single zigzag traversal (passing through the street segment once) or of servicing the two sides separately by two traversals. Fourth, the so-called nonrequired street segments may be used to get from one point to another. All street segments may be traversed in both directions for deadheading. Again, different directions of traversal of a segment may incur different costs. Moreover, the different traversal modes (one-side service, zigzag service, deadheading) may incur different costs.

In Irnich [40], an integer programming formulation for the WRPPZ is presented. The author also proposes a procedure to transform the WRPPZ into an ATSP, which is later transformed into a TSP and solved using Concorde (Applegate et al. [2]). From a theoretical point of view, it is clear that the additional flexibility provided by the zigzagging option enables better feasible solutions to be build. The computational results obtained indicate that the extent of improvement depends on the cost of services and on the distribution of the types of service edges.

The same author presents an extended version of the WRPPZ, called EXT-WRPP, in Irnich [41]. In addition to the WRPPZ, the new problem considers several relevant practical aspects such as turn restrictions, clustered street segments which must be serviced consecutively, public transport for reaching districts served on foot, and open tours. The author points out that the EXT-WRPP is of great practical importance for mail delivery, since in Germany there are more than 50,000 districts to be serviced every day, each district corresponding to an EXT-WRPP. For solving this problem, a transformation into an asymmetric or symmetric TSP is proposed. Since the computational experiments show that the solution of the resulting TSP instances with standard algorithms does not work satisfactorily, a new heuristic method is proposed in the paper that outperforms standard TSP heuristics with respect to solution quality and computing time.

Acknowledgments

The authors would like to thank the Spanish Ministerio of Economía y Competitividad (project MTM2012-36163-C06-02) and the Generalitat Valenciana (project GVPROMETEO2013-049) for their support.

Bibliography

[1] H.M. Afsar, N. Jozefowiez, and P. Lopez, *A branch-and-price algorithm for the windy rural postman problem*, RAIRO Operations Research, 45 (2011), pp. 353–364.

[2] D. Applegate, R.E. Bixby, V. Chvátal, and W. Cook, *Concorde*, 1999. Available at http://www.math.princeton.edu/tsp/concorde.html.

[3] T. Ávila, *Algunos problemas de rutas por arcos*, Ph.D. thesis, University of Valencia, Valencia, Spain, 2014.

[4] T. Ávila, Á. Corberán, I. Plana, and J.M. Sanchis, *An ILS-based metaheuristic for the stacker crane problem*, in Evolutionary Computation in Combinatorial Optimization, J.K. Hao and M. Middendorf, eds., Lecture Notes in Computer Science 7245, Springer, Berlin, 2012, pp. 25–36.

[5] ——, *The stacker crane problem and the directed general routing problem*, Technical report, Department of Statistics and Ops. Res., University of Valencia, Valencia, Spain, 2013.

[6] ——, *A new branch-and-cut algorithm for the generalized directed rural postman problem*, Technical report, Department of Statistics and Ops. Res., University of Valencia, Valencia, Spain, 2013.

[7] C. Balaguer, A. Giménez, J.M. Pastor, V.M. Padrón, and M. Abderrahim, *A climbing autonomous robot for inspection applications in 3D complex environments*, Robotica, 18 (2000), pp. 287–297.

[8] E. Benavent, A. Carrotta, Á. Corberán, J.M. Sanchis, and D. Vigo, *Lower bounds and heuristics for the windy rural postman problem*, European Journal of Operational Research, 176 (2007), pp. 855–869.

[9] E. Benavent, Á. Corberán, E. Piñana, I. Plana, and J.M. Sanchis, *New heuristic algorithms for the windy rural postman problem*, Computers & Operations Research, 32 (2005), pp. 3111–3128.

[10] E. Benavent and D. Soler, *The directed rural postman problem with turn penalties*, Transportation Science, 33 (1999), pp. 408–418.

[11] G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte, *Static pickup and delivery problems: A classification scheme and survey*, TOP, 15 (2007), pp. 1–31.

[12] M. Blais and G. Laporte, *Exact solution of the generalized routing problem through graph transformations*, The Journal of the Operational Research Society, 54 (2003), pp. 906–910.

[13] G. Carpaneto, M. Dell'Amico, and P. Toth, *Exact solution of large scale, asymmetric traveling salesman problems*, ACM Transactions on Mathematial Software, 21 (1995), pp. 395–409.

[14] N. Christofides, *Worst-case analysis of a new heuristic for the traveling salesman problem*, Technical report, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA, 1976.

[15] N. CHRISTOFIDES, E. BENAVENT, V. CAMPOS, Á. CORBERÁN, AND E. MOTA, *An optimal method for the mixed postman problem*, in System Modelling and Optimization, P. Thoft-Christensen, ed., Lecture Notes in Control and Information Sciences 59, Springer, Berlin, 1984, pp. 641–649.

[16] N. CHRISTOFIDES, V. CAMPOS, Á. CORBERÁN, AND E. MOTA, *An algorithm for the rural postman problem on a directed graph*, Mathematical Programming Study, 26 (1986), pp. 155–166.

[17] C.C. CHYU, *A mixed strategy heuristic for the mixed arc routing problem*, Journal of the Chinese Institute of Industrial Engineers, 18 (2001), pp. 68–76.

[18] J. CIRASELLA, D.S. JOHNSON, L.A. MCGEOCH, AND W. ZHANG, *Traveling salesman problem: Algorithms, instance generators, and tests*, in ALENEX 2001 Proceedings, A. L. Buchsbaum and J. Snoeyink, eds., Lecture Notes in Computer Science 2153, Springer, Berlin, 2001, pp. 32–59.

[19] Á. CORBERÁN, R. MARTÍ, E. MARTÍNEZ, AND D. SOLER, *The rural postman problem on mixed graphs with turn penalties*, Computers & Operations Research, 29 (2002), pp. 887–903.

[20] Á. CORBERÁN, R. MARTÍ, AND A. ROMERO, *Heuristics for the mixed rural postman problem*, Computers & Operations Research, 27 (2000), pp. 183–203.

[21] Á. CORBERÁN, G. MEJÍA, AND J.M. SANCHIS, *New results on the mixed general routing problem*, Operations Research, 53 (2005), pp. 363–376.

[22] Á. CORBERÁN, E. MOTA, AND J.M. SANCHIS, *A comparison of two different formulations for arc routing problems on mixed graphs*, Computers & Operations Research, 33 (2006), pp. 3384–3402.

[23] Á. CORBERÁN, M. OSWALD, I. PLANA, G. REINELT, AND J.M. SANCHIS, *New results on the windy postman problem*, Mathematical Programming, 132 (2012), pp. 309–332.

[24] Á. CORBERÁN, I. PLANA, AND J.M. SANCHIS, *Zigzag inequalities: A new class of facet-inducing inequalities for arc routing problems*, Mathematical Programming, 108 (2006), pp. 79–96.

[25] ——, *A branch & cut algorithm for the windy general routing problem and special cases*, Networks, 49 (2007), pp. 245–257.

[26] ——, *The windy general routing polyhedron: A global view of many known arc routing polyhedra*, SIAM Journal on Discrete Mathematics, 22 (2008), pp. 606–628.

[27] Á. CORBERÁN, A. ROMERO, AND J.M. SANCHIS, *The mixed general routing polyhedron*, Mathematical Programming, 96 (2003), pp. 103–137.

[28] Á. CORBERÁN AND J.M. SANCHIS, *A polyhedral approach to the rural postman problem*, European Journal of Operational Research, 79 (1994), pp. 95–114.

[29] ——, *The general routing problem polyhedron: Facets from the RPP and GTSP polyhedra*, European Journal of Operational Research, 108 (1998), pp. 538–550.

[30] M. DREXL, *On Some Generalized Routing Problems*, Ph.D. thesis, Aachen University, Aachen, Germany, 2007.

[31] ——, *On the generalized directed rural postman problem*, The Journal of the Operational Research Society, 65 (2014), pp. 1143–1154.

[32] M. DROR AND A. LANGEVIN, *A generalized traveling salesman problem approach to the directed clustered rural postman problem*, Transportation Science, 31 (1997), pp. 187–192.

[33] H.A. EISELT, M. GENDREAU, AND G. LAPORTE, *Arc routing problems, part 2: The rural postman problem*, Operations Research, 43 (1995), pp. 399–414.

[34] G.N. FREDERICKSON, *Approximation algorithms for some postman problems*, Journal of the ACM, 26 (1979), pp. 538–554.

[35] G.N. FREDERICKSON, M.S. HECHT, AND C.E. KIM, *Approximation algorithms for some routing problems*, SIAM Journal on Computing, 7 (1978), pp. 178–193.

[36] M-H. HÀ, N. BOSTEL, A. LANGEVIN, AND L-M. ROUSSEAU, *An exact algorithm for close enough traveling salesman problem*, in Proceedings of the 1st International Conference on Operations Research and Enterprise Systems, 2012, pp. 233–238.

[37] ——, *Solving the close enough arc routing problem*, Networks, 63 (2014), pp. 107–118.

[38] R. HASSIN AND S. KHULLER, *z-approximations*, Journal of Algorithms, 41 (2001), pp. 429–442.

[39] A. HERTZ, G. LAPORTE, AND P. NANCHEN-HUGO, *Improvement procedures for the undirected rural postman problem*, INFORMS Journal on Computing, 11 (1999), pp. 53–62.

[40] S. IRNICH, *A note on postman problems with zigzag service*, INFOR, 43 (2005), pp. 33–39.

[41] ——, *Solution of real-world postman problems*, European Journal of Operational Research, 190 (2008), pp. 52–67.

[42] P. LACOMME, C. PRINS, AND W. RAMDANE-CHÉRIF, *A genetic algorithm for the capacitated arc routing problem and its extensions*, in Applications of Evolutionary Computing, E.J.W. Boers, ed., Lecture Notes In Computer Science 2037, Springer-Verlag, Berlin, 2001, pp. 473–483.

[43] ——, *Fast algorithms for general arc routing problems*, in IFORS 2002, Edinburgh, UK, 2002.

[44] G. LAPORTE, *Modeling and solving several classes of arc routing problems as traveling salesman problems*, Computers & Operations Research, 24 (1997), pp. 1057–1061.

[45] J.K. LENSTRA AND A. RINNOOY KAN, *On the general routing problem*, Networks, 6 (1976), pp. 273–280.

[46] A.N. LETCHFORD, *New inequalities for the general routing problem*, European Journal of Operational Research, 96 (1997), pp. 317–322.

[47] C.E. NOON AND J.C. BEAN, *A Lagrangian based approach for the asymmetric generalized traveling salesman problem*, Operations Research, 39 (1991), pp. 623–632.

[48] I. OR, *Traveling Salesman-Type Combinatorial Problems and Their Relation to the Logistics of Regional Blood Banking*, Ph.D. thesis, Northwestern University, Evanston, IL, 1976.

[49] C.S. ORLOFF, *A fundamental problem in vehicle routing*, Networks, 4 (1974), pp. 35–64.

[50] I. PLANA, *The Windy General Routing Problem*, Ph.D. thesis, University of Valencia, Valencia, Spain, 2005.

[51] B. RAGHAVACHARI AND J. VEERASAMY, *A 3/2 approximation algorithm for the mixed postman problem*, SIAM Journal on Discrete Mathematics, 12 (1999), pp. 425–433.

[52] R. SHUTTLEWORTH, B.L. GOLDEN, S. SMITH, AND E. WASIL, *Advances in meter reading: Heuristic solution of the close enough traveling salesman problem over a street network*, in The Vehicle Routing Problem: Lastest Advances and New Challenges, B.L. Golden, S. Raghavan, and E. Wasil, eds., Springer, Berlin, 2008, pp. 487–501.

[53] D. SOLER, E. MARTÍNEZ, AND J. MICÓ, *A transformation for the mixed general routing problem with turn penalties*, The Journal of the Operational Research Society, 59 (2008), pp. 540–547.

[54] J. SROUR AND S. VAN DE VELDE, *Are stacker crane problems easy? A statistical study*, Computers & Operations Research, 40 (2013), pp. 674–690.

[55] Z. WIN, *On the windy postman problem on Eulerian graphs*, Mathematical Programming, 44 (1989), pp. 97–112.

[56] L. ZHANG, *Polynomial algorithms for the k-Chinese postman problem*, in Proceedings of the IFIP 12th World Computer Congress. Volume 1: Algorithms, Software, Architecture, J. Van Leeuwen, ed., Elsevier, Amsterdam, 1992, pp. 430–435.

[57] L. ZHANG AND W. ZHENG, *Genetic coding for solving both the stacker crane problem and its k-variant*, in Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, IEEE, Piscataway, NJ, 1995, pp. 1061–1066.