Mathematical Programming Approaches to the Traveling Salesman Problem

Adam N. Letchford Andrea Lodi

The *Traveling Salesman Problem* or TSP is probably the best known combinatorial optimisation problem. Informally speaking, one is given a set of cities, along with the cost of traveling between each pair of cities, and one wishes to find a tour of all the cities of minimum cost. The TSP has applications, not only in OR/MS, but in many other fields. In fact, there are no fewer than four books devoted to it [1, 23, 28, 40].

The TSP is notoriously hard to solve, being one of Karp's original $\mathcal{NP}$-complete problems [27]. Nevertheless, and surprisingly, large-scale instances arising in practice can often be solved to proven optimality (or near-optimality) by sophisticated mathematical programming techniques. This article gives an introduction to these techniques.

The article is structured as follows. In Section 1, we review some of the basic mathematical programming formulations of the TSP. In Section 2, we discuss polyhedral theory, which is used to derive stronger formulations. In Section 3, we discuss separation routines, which are used to strengthen formulations iteratively. Finally, in Section 4 we discuss exact algorithms for the TSP based on such strengthened formulations.

We remark that some other effective techniques for tackling the TSP exist. For an introduction to *combinatorial* TSP algorithms (i.e., algorithms that work with graph-theoretic relaxations rather than linear programming relaxations), see entry #1.3.5.2. For heuristic and meta-heuristic approaches (which give good, but not necessarily optimal solutions), see entry #1.3.5.3.

Throughout this article, we distinguish between the *symmetric* TSP (STSP), in which the cost of travelling from city A to city B is the same as the cost of travelling in the reverse direction, and the *asymmetric* TSP (ATSP), in which these costs are permitted to be different.

1 Formulations

Combinatorial optimisation problems can often be formulated in several different ways — see entry #1.4.1.1. In the case of the STSP, the standard formulation is the following, due to Dantzig, Fulkerson & Johnson [14]. We are given a complete undirected graph G, with node set V and edge set E.

For each $e \in E$, we let c_e denote the cost of traversing edge e. For any node set $S \subset V$, we let $\delta(S)$ denote the set of edges with exactly one end-node in S. Next, for each edge $e \in E$, we define a binary variable x_e, taking the value 1 if and only if edge e is in the tour. Finally, for any set of edges $F \subset E$, we let $x(F)$ denote $\sum_{e \in F} x_e$. The STSP is then equivalent to the following 0-1 Linear Program (0-1 LP):

$$
\begin{array}{rll}
\min & \sum_{e \in E} c_e x_e & \\
\text{s.t.} & x(\delta(\{i\})) = 2 \quad (\forall i \in V) & (1) \\
& x(\delta(S)) \geq 2 \quad (\forall S \subset V : 2 \leq |S| \leq |V|/2) & (2) \\
& x \in \{0,1\}^{|E|}. & (3)
\end{array}
$$

The equations (1), called *degree equations*, simply express the fact that the salesman must arrive at and depart from each city. The inequalities (2), called *subtour elimination constraints* (SECs), ensure that the tour is connected.

The ATSP can be formulated in a similar way. Now we have a complete directed graph G, with node set V and arc set A. For each $a \in A$, we let c_a denote the cost of traversing arc a. For any set $S \subset V$, we let $\delta^+(S)$ denote the set of arcs going from S to $V \setminus S$, and $\delta^-(S)$ denote the set of arcs going in the reverse direction. For each arc $a \in A$, we have a binary variable x_a. The ATSP is then equivalent to the following 0-1 LP:

$$
\begin{array}{rll}
\min & \sum_{a \in A} c_a x_a & \\
\text{s.t.} & x(\delta^+(\{i\})) = 1 \quad (\forall i \in V) & (4) \\
& x(\delta^-(\{i\})) = 1 \quad (\forall i \in V) & (5) \\
& x(\delta^+(S)) \geq 1 \quad (\forall S \subset V : 2 \leq |S| \leq |V|/2) & (6) \\
& x \in \{0,1\}^{|A|}. & (7)
\end{array}
$$

The equations (4) and (5) are called *out-degree* and *in-degree* equations, respectively. The inequalities (6) are again called SECs.

Note that, in both formulations, the SECs are exponential in number. It is in fact possible to formulate the STSP and ATSP as 0-1 LPs in which the number of constraints is polynomial, provided that one is willing to introduce a polynomial number of additional variables. An extensive survey of such alternative formulations was recently given by Öncan *et al.* [35]. Nevertheless, the formulations given here have been found to be very useful in practice, and are therefore the most commonly used. The large number of SECs is not a serious problem, since one can generate the SECs dynamically as cutting planes (see entry #1.4.1.5).

2 Polyhedral Theory

In this section, we review some of the known polyhedral results for the TSP. We cover the STSP in Subsection 2.1 and the ATSP in Subsection 2.2. We assume that the reader is already familiar with the polyhedral approach to combinatorial optimisation problems, which is covered in entry #1.4.3.1.

2.1 The symmetric traveling salesman polytope

The convex hull in $\mathbb{R}^{|E|}$ of solutions to the system (1)-(3), for a given number n of nodes, is called the *symmetric traveling salesman polytope of order* n, and is denoted by STSP(n). The first systematic study of STSP(n) was performed by Grötschel & Padberg [21], who proved the following results:

- The degree equations (1) give a complete and non-redundant description of the affine hull of STSP(n).
- The non-negativity inequalities $x_e \geq 0$, for all $e \in E$, define facets of STSP(n) for $n \geq 4$.
- The SECs (2) define facets of STSP(n) for $n \geq 4$.
- Let H be a subset of V with $3 \leq |H| \leq n-2$ and let $F \subset \delta(H)$ be a set of node-disjoint edges with $|F| \geq 3$ and odd. The *blossom* inequality $x(\delta(H) \setminus F) \geq x(F) - |F| + 1$ defines a facet of STSP(n).

Grötschel & Padberg [21] also proved a 'lifting' theorem that enables one to convert simple facets into more complex facets. When lifting is applied to the blossom inequalities, one obtains the following *comb* inequalities:

$$x(\delta(H)) + \sum_{j=1}^{t} x(\delta(T_j)) \geq 3t + 1.$$

Here, $t \geq 3$ is an odd integer, H is a proper subset of V (called the *handle*), and $T_1, \ldots, T_j$ are disjoint node-sets (called the *teeth*), such that $T_j \cap H \neq \emptyset$ and $T_j \setminus H \neq \emptyset$ for $j = 1, \ldots, t$.

The comb inequalities are undoubtedly a very important class of inequalities for the STSP. They are used in all branch-and-cut algorithms of which we are aware.

Since the publication of [21], the polytope STSP(n) has been studied in great depth. Several generalisations of the comb inequalities are known, with exotic names like *clique-tree*, *star*, *hyperstar*, *bipartition* and *binested* inequalities. There are also several other classes of facet-inducing inequalities known, such as *chain*, *ladder*, *crown* and *hypohamiltonian* inequalities. For the sake of space, we do not review these developments here, and refer the reader to the excellent surveys by Grötschel & Padberg [22], Jünger *et al.* [26] and Naddef [32]. We will however mention some of the inequalities again, in subsequent sections.

2.2 The asymmetric traveling salesman polytope

The convex hull in $\mathbb{R}^{|A|}$ of solutions to the system (4)-(7), again for a given n, is called the *asymmetric traveling salesman polytope of order* n, and is denoted by ATSP(n). A survey of early work on ATSP(n) was given by Grötschel & Padberg [22]. Some of the key early results are as follows:

- The out- and in-degree equations (4)–(5) define the affine hull of ATSP(n), and all but one of them (arbitrarily chosen) are linearly independent.
- The non-negativity inequalities $x_a \geq 0$, for all $a \in A$, define facets of ATSP(n) for $n \geq 4$.
- Any valid inequality for STSP(n) can be converted into a valid inequality for ATSP(n), by simply replacing x_e with $x_{ij} + x_{ji}$ for all edges $e = (i,j) \in E$. The resulting inequalities are called *symmetric.*
- The SECs (6), which are symmetric, define facets of ATSP(n) for $n \geq 4$.

Grötschel & Padberg [22] also list various *asymmetric* inequalities. Among them, we mention the so-called D_k^+ and D_k^- inequalities, which have proven to be fairly useful in practice. The D_k^+ inequalities take the form:

$$x_{i_1 i_k} + \sum_{h=2}^{k} x_{i_h i_{h-1}} + 2\sum_{h=2}^{k-1} x_{i_1 i_h} + \sum_{h=3}^{k-1} x(\{i_2, \ldots, i_{h-1}\}, i_h) \leq k-1, \tag{8}$$

where $(i_1, \ldots, i_k)$ is any sequence of $k \in \{3, \ldots, n-1\}$ distinct nodes. The D_k^- inequalities are derived from the D_k^+ inequalities by simply swapping the left-hand side coefficients of the two arcs (i,j) and (j,i), for all $i,j \in V$, $i < j$. This is a perfectly general operation, called *transposition* in [22].

Another important class of inequalities, which turn out to be very useful in practice, were introduced by Balas [2]. Two distinct arcs (i,j) and (u,v) are called *incompatible* if $i = u$, or $j = v$, or $i = v$ and $j = u$; *compatible* otherwise. A *Closed Alternating Trail* (CAT) is a sequence $T = \{a_1, \ldots, a_t\}$ of t distinct arcs such that, for $k = 1, \ldots, t$, arc a_k is incompatible with arcs a_{k-1} and a_{k+1}, and compatible with all other arcs in T (with $a_0 := a_t$ and $a_{t+1} := a_1$). Now, call a node v a *source* if $|\delta^+(v) \cap T| = 2$, and a *sink* if $|\delta^-(v) \cap T| = 2$. Note that a node can be a source and a sink simultaneously. Let Q be the set of arcs $(i,j) \in A \setminus T$ such that i is a source and j is a sink. Given a CAT T with t odd, the *odd CAT* inequality $x(T \cup Q) \leq \lfloor t/2 \rfloor$ is valid for ATSP(n). Balas [2] proved that odd CAT inequalities define facets of ATSP(n) under mild conditions.

There exist several other papers on ATSP(n). Again, for the sake of brevity, we do not review them all here, and refer the reader to the excellent survey [3].

3 Separation Routines

In order to use a class of valid inequalities as cutting planes, one needs a *separation routine*, i.e., a routine for detecting when an inequality in that class is violated by a given LP solution x^* (see entry #1.4.1.5). As usual, we distinguish between *exact* separation routines, which find a violated inequality in that class whenever one exists, and *heuristic* separation routines, which may fail to find a violated inequality.

3.1 Separation routines for the STSP

Several positive results are known concerning the complexity of exact separation for various inequalities:

- As noted by Hong [24], SEC separation amounts to a minimum weight cut problem, and is therefore solvable in polynomial time. (The current fastest deterministic minimum weight cut algorithm is that of Nagamochi *et al.* [34].)
- The separation problem for the blossom inequalities can be solved in polynomial time (Padberg & Rao [36]). The current fastest algorithm is that of Letchford *et al.* [31].
- There exists a polynomial-time separation algorithm for a class of inequalities that includes all blossom inequalities and a large subclass of the comb inequalities (Fleischer *et al.* [17]).
- There is a polynomial-time algorithm that generates a violated valid inequality whenever a comb inequality is violated by 1 (Caprara *et al.* [6]).
- There exists a polynomial-time separation algorithm for a class of inequalities that includes all comb inequalities with a fixed handle H (Caprara & Letchford [7]).
- The separation problem for bipartition inequalities (and therefore comb and clique-tree inequalities) can be solved in polynomial time, provided the number of teeth is bounded by a constant (Carr [8]).
- More generally, there exists a polynomial-time separation algorithm for any class of inequalities that can be obtained by 'lifting' an inequality that defines a facet of STSP(n), for constant n (Carr [9]).

Further positive results are known for the case in which the *support graph* is planar. (The support graph is the graph induced by the edges with positive x^*-value.) Fleischer & Tardos [18] found an $\mathcal{O}(n^2 \log n)$ algorithm that produces a violated comb inequality whenever a comb inequality is violated by 1, and Letchford [29] gave an $\mathcal{O}(n^3)$ exact separation algorithm for a generalisation of the comb inequalities, called 'domino-parity' inequalities. Further results for the planar case can be found in [30].

In addition to these exact separation results, there exists a wide range of separation heuristics. For example:

- Crowder & Padberg [13] presented a fast heuristic for SECs, that is based on the idea of iteratively 'shrinking' the support graph by identifying certain pairs of nodes.
- Grötschel & Holland [20] proposed a heuristic for comb separation, in which one first shrinks the support graph, and then calls an exact blossom separation algorithm on the shrunk support graph.
- Padberg & Rinaldi [37] presented several separation heuristics for SECs, blossom, comb and clique tree inequalities, along with some new conditions under which shrinking can be safely performed.
- Naddef & Thienel [33] presented new heuristics for comb, clique-tree, path, bipartition and ladder inequalities. They are based on shrinking and local search.
- Applegate *et al.* [1] analysed in depth certain data structures that can be used to speed up various SEC, blossom and comb heuristics.
- Boyd *et al.* [4] and Cook *et al.* [12] presented heuristics for domino-parity inequalities, based on shrinking the support graph to make it planar, and then running the algorithm of Letchford [29].

Finally, we mention two 'non-standard' heuristics for separation: the 'small instance' approach of Christof & Reinelt [10] and the 'local cuts' approach of Applegate *et al.* [1]. Both of them begin by shrinking the support graph down to a tiny size. The small instance approach then searches through a list of inequality 'templates', whereas the local cuts approach solves a series of tiny STSP instances to generate a cutting plane.

3.2 Separation routines for the ATSP

In Subsection 2.2, we mentioned that every valid inequality for STSP(n) can be converted into a valid inequality for ATSP(n), simply by replacing x_e with $x_{ij} + x_{ji}$ for all edges $e = (i,j) \in E$. This implies that every separation algorithm for the STSP can be used, as a "black box", for the ATSP as well. To this end, given the ATSP (fractional) point $x^* \in [0,1]^{|A|}$, one first defines the undirected counterpart $\tilde{x} \in [0,1]^{|E|}$ by means of the transformation

$$\tilde{x}_e := x^*_{ij} + x^*_{ji} \qquad \text{for all edges } e = (i,j) \in E,$$

and then applies the STSP separation algorithm to $\tilde{x}$. On return, the detected most violated STSP inequality is transformed into its ATSP counterpart, both inequalities having the same degree of violation.

The above approach to separation can be used only for *symmetric* ATSP inequalities. The only paper of which we are aware that addresses separation for *asymmetric* ATSP inequalities is Fischetti & Toth [16]. These authors presented an exact separation algorithm for D_k^+ and D_k^- inequalities, based on partial enumeration of suitable node sequences $(i_1, \ldots, i_k)$, for $k = 2, \ldots, n-1$. To prune the search, upper bounds on the violation of the inequalities are used. The authors also presented a heuristic separation algorithm for odd CAT inequalities, based on the computation of minimum weight odd circuits in an auxilary 'incompatibility graph'. This separation algorithm can be viewed as a specialized version of a scheme proposed by Caprara & Fischetti [5] for the separation of a subclass of Chvátal-Gomory cuts for general integer programming problems.

4 Exact TSP Algorithms

In this section, we review the main exact algorithms for the TSP that are based on the use of strong valid inequalities as cutting planes. We assume that the reader is already familiar with the basic ideas of branch-and-bound and branch-and-cut, which are covered in entries #1.4.1.2 and #1.4.1.5, respectively.

4.1 Exact algorithms for the STSP

The first paper to propose the use of Linear Programming to solve the STSP was Dantzig *et al.* [14] in 1954. The authors computed an optimal tour through 49 cities in the United States, an impressive achievement at the time. The cutting planes, mainly SECs, were identified by eye. Moreover, the authors used a limited form of branching, even though branch-and-bound had not been invented at the time.

In the 1970s and 1980s, several papers appeared in which larger TSP instances were solved by the same approach, but with the addition of other kinds of cutting planes, such as blossom and comb inequalities. A good example is Grötschel [19], which details the solution of a 120-city problem.

The next step was the development of what is now called the 'cut-and-branch' approach, in which a cutting plane algorithm is used to create a strong LP relaxation, which is then fed into a branch-and-bound solver. An early example can be found in Crowder & Padberg [13]. This approach was pushed to its limits by Grötschel & Holland [20], who solved instances with up to around 600 cities.

A disadvantage of the cut-and-branch approach is that the integer solution obtained might violate some SECs, in which case they must be added to the formulation, and branch-and-bound must be repeated. The natural solution to this is to run separation algorithms at every node of the branch-and-bound tree. This is the *branch-and-cut* approach of Padberg & Rinaldi [38]. Using this approach, Padberg & Rinaldi solved instances with up to around 2000 cities, using SECs and blossom, comb and clique-tree inequalities.

The branch-and-cut approach has been further refined, for example by Naddef & Thienel [33], Applegate *et al.* [1] and Cook *et al.* [12]. The software of Applegate *et al.*, called `Concorde`, has been made freely available on the web [11]. It is capable of solving instances with tens of thousands of cities to proven optimality. According to Cook *et al.* [12], the domino-parity inequalities and the local cuts are the most effective cutting planes for large instances.

4.2 Exact algorithms for the ATSP

Only two papers have been published concerned with branch-and-cut algorithms for the ATSP: Fischetti & Toth [16] and Fischetti *et al.* [15].

The algorithm of Fischetti & Toth [16] is based closely on the Padberg & Rinaldi [38] framework. There are, however, some important details:

1. A sophisticated scheme is used for pricing (column generation), which exploits the fact that a directed tour cannot select arcs with negative reduced cost in an arbitrary way.

2. Asymmetric inequalities — D_k^+, D_k^- and odd CAT inequalities — are used in addition to symmetric ones.

3. Rounds of violated inequalities are added when possible, rather than only the most violated one.

4. A primal heuristic, based on detecting Hamiltonian cycles in the (directed) support graph, is extensively used.

The resulting algorithm solved to optimality most of the ATSP instances in the literature.

Fischetti *et al.* [15] proposed some enhancements to the Fischetti-Toth algorithm. In particular, the branching variable in [16] was selected among those closest to 0.5 (a classical criterion) while in [15] priority for branching was given to the variables that have been persistently fractional in the last (consecutive) LP optimal solutions at any branching node.

Fischetti *et al.* [15] also tested a rather different approach, in which the ATSP instance is transformed to an STSP instance, which is then solved by the STSP solver `Concorde` mentioned in the previous subsection. They tested two different transformations: the one due to Karp [27], in which an ATSP instance with n nodes is converted to an STSP instance with $3n$

nodes, and the one due to Jonker & Volgenant [25], in which the resulting STSP instance has only $2n$ nodes. The computational results in [15] show that this alternative approach is competitive with (although not superior to) the approach in [16].

We close this discussion by pointing out an important difference between the STSP and the ATSP. In the case of the STSP, branch-and-cut is currently the undisputed best approach for solving large-scale instances. This is not so with the ATSP: for some instances, such as randomly generated instances with no correlation between c_{ij} and c_{ji}, combinatorial algorithms based on *assignment relaxations* can perform as well or even better. This is discussed in more detail in entry #1.3.5.2.

References

[1] Applegate, D, Bixby, RE, Chvátal, V and Cook, W. The Traveling Salesman Problem: a Computational Study. 2007. Princeton University Press, Princeton.

[2] Balas, E. The asymmetric assignment problem and some new facets of the traveling salesman polytope on a directed graph. SIAM J. Discr. Math. 1989; 2: 425–451.

[3] Balas, E, Fischetti, M. Polyhedral theory for the asymmetric traveling salesman problem. Pp 117–168 in Gutin & Punnen (eds), *op. cit.*

[4] Boyd, SC, Cockburn, S, Vela, D. On the domino-parity inequalities for the TSP. Math. Program. 2007; 110: 501–519.

[5] Caprara, A, Fischetti, M. $\{0, \frac{1}{2}\}$-Chvátal-Gomory cuts. Math. Program. 1996; 74: 221–236.

[6] Caprara, A, Fischetti, M, Letchford, AN. On the separation of maximally violated mod-k cuts. Math. Program. 2000; 87: 37–56.

[7] Caprara, A, Letchford, AN. On the separation of split cuts and related inequalities. Math. Program. 2003; 94: 279–294.

[8] Carr, RD. Separating clique tree and bipartition inequalities having a fixed number of handles and teeth in polynomial time. Math. Oper. Res. 1997; 22: 257–265.

[9] Carr, RD. Separation algorithms for classes of STSP inequalities arising from a new STSP relaxation. Math. Oper. Res. 2004; 29: 80–91.

[10] Christof, T, Reinelt, G. Combinatorial optimization and small polytopes. TOP 1996; 4: 1–53.

[11] Concorde TSP Solver. Available at:
`http://www.tsp.gatech.edu/concorde.html`

[12] Cook, W, Espinoza, D, Goycoolea, M. Computing with domino-parity inequalities for the TSP. INFORMS J. Comput. 2007; 19: 356–365.

[13] Crowder, H, Padberg, MW. Solving large-scale symmetric traveling salesman problems to optimality. Mgmt. Sci. 1980; 26: 495–509.

[14] Dantzig, GB, Fulkerson, DR, Johnson, SM. Solution of a large-scale traveling salesman problem. Oper. Res. 1954; 2: 393–410.

[15] Fischetti, M, Lodi, A, Toth, P. Exact methods for the asymmetric traveling salesman problem. Pp 169–205 in Gutin & Punnen (eds), *op. cit.*

[16] Fischetti, M, Toth, P. A polyhedral approach to the asymmetric traveling salesman problem. Mgmt. Sci. 1997; 43: 1520–1536.

[17] Fleischer, LK, Letchford, AN, Lodi, A. Polynomial-time separation of a superclass of simple comb inequalities. Math. Oper. Res. 2006; 31: 696–713.

[18] Fleischer, LK, Tardos É. Separating maximally violated comb inequalities in planar graphs. Math. Oper. Res. 1999; 24: 130–148.

[19] Grötschel, M. On the symmetric travelling salesman problem: solution of a 120–city problem. Math. Program. Study 1980; 12: 61–77.

[20] Grötschel, M, Holland, O. Solution of large-scale symmetric traveling salesman problems. Math. Program. 1991; 51: 141–202.

[21] Grötschel, M, Padberg, MW. On the symmetric travelling salesman problem (parts I and II). Math. Program. 1979; 16: 265–302.

[22] Grötschel, M, Padberg, MW. Polyhedral theory. Pp 251–305 in Lawler *et al.* (eds), *op. cit.*

[23] Gutin, G and Punnen, AP, eds. The Traveling Salesman Problem and its Variations. 2002. Kluwer, Dortrecht.

[24] Hong, S. A Linear Programming Approach for the Traveling Salesman Problem. PhD Thesis, Johns Hopkins University, Baltimore, Maryland, USA, 1972.

[25] Jonker, R, Volgenant, T. Transforming asymmetric into symmetric traveling salesman problems. Oper. Res. Lett. 1983; 2: 161–163.

[26] Jünger, M, Reinelt, G and Rinaldi, G. The traveling salesman problem. Pp 225–330 in M Ball, T Magnanti, C Monma and G Nemhauser, eds. Network Models. 1995. Elsevier, Amsterdam.

[27] Karp, RM. Reducibility among combinatorial problems. Pp 85-103 in RE Miller and JW Thatcher, eds. Complexity of Computer Computations. 1972. Plenum, New York.

[28] Lawler, EL, Lenstra, JK, Rinnooy Kan AHG and Shmoys, DB, eds. The Traveling Salesman Problem. 1985. Wiley, Chichester.

[29] Letchford, AN. Separating a superclass of comb inequalities in planar graphs. Math. Oper. Res. 2000; 25: 443–454.

[30] Letchford, AN, Pearson, NA. Exploiting planarity in separation routines for the symmetric traveling salesman problem. Discr. Opt. 2008; 5: 220–230.

[31] Letchford, AN, Reinelt, G, Theis, DO. Odd minimum cut-sets and b-matchings revisited. SIAM J. Discr. Math. 2008; 22: 1480–1487.

[32] Naddef, D. Polyhedral theory and branch-and-cut algorithms for the TSP. Pp 29–114 in Gutin & Punnen (eds), *op. cit.*

[33] Naddef, D, Thienel, S. Efficient separation routines for the symmetric traveling salesman problem (parts I and II). Math. Program. 2002; 92: 237–283.

[34] Nagamochi, H, Ono, T, Ibaraki, T. Implementing an efficient minimum capacity cut algorithm. Math. Program. 1994; 67: 325–341.

[35] Öncan, T, Altinel, IK, Laporte, G. A comparative analysis of several asymmetric traveling salesman problem formulations. Comput. & Oper. Res. 2009; 36: 637–654.

[36] Padberg, MW, Rao, MR. Odd minimum cut-sets and b-matchings. Math. Oper. Res. 1982; 7: 67–80.

[37] Padberg, MW, Rinaldi, G. Facet identification for the symmetric traveling salesman polytope. Math. Program. 1990; 47: 219–257.

[38] Padberg, MW, Rinaldi, G. A branch-and-cut algorithm for the resolution of large-scale symmetric travelling salesman problems. SIAM Rev. 1991; 33: 60–100.

[39] Reinelt, G. TSPLIB — a traveling salesman problem library. ORSA J. Comput. 1991; 3: 376–384.

[40] Reinelt, G. The Traveling Salesman: Computational Solutions for TSP Applications. 1994. Springer, Berlin.

Author Addresses:

Adam N. Letchford
Department of Management Science
Lancaster University
Lancaster LA1 4YX
United Kingdom.
E-mail: `a.n.letchford@lancaster.ac.uk`

Andrea Lodi
DEIS,
University of Bologna
Viale Risorgimento, 2
40136 Bologna
Italy.
E-mail: `andrea.lodi@unibo.it`

Abstract: The *Traveling Salesman Problem* or TSP is a fundamental and well-known problem in combinatorial optimisation. At present, the most successful algorithms for solving large-scale instances of the TSP to proven (near-)optimality are based on integer programming. This entry introduces the main theoretical and algorithmic tools involved. Topics covered include: formulations of the TSP, polyhedral theory, separation routines, exact TSP algorithms.

Key Words: traveling salesman problem — integer programming — polyhedral combinatorics.