Discrete Optimization 21 (2016) 25–41

Contents lists available at ScienceDirect

Discrete Optimization

www.elsevier.com/locate/disopt

Polyhedral results and a branch-and-cut algorithm for the double traveling Salesman problem with multiple stacks

Michele Barbato, Roland Grappe, Mathieu Lacroix, Roberto Wolfler Calvo*

Université Paris 13, Sorbonne Paris Cité, LIPN, CNRS (UMR 7030), F-93430, Villetaneuse, France

ARTICLE INFO

Article history:
Received 1 August 2015
Received in revised form 25 April 2016
Accepted 26 April 2016
Available online 5 June 2016

MSC:
90B06
90C10
90C27
90C57

Keywords:
Traveling Salesman problem
Multiple stacks
Polyhedral study
Branch-and-cut

ABSTRACT

In the double TSP with multiple stacks, one performs a Hamiltonian circuit to pick up n items, storing them in a vehicle with s stacks of finite capacity q satisfying last-in-first-out constraints, and then delivers every item by performing a Hamiltonian circuit. We introduce an integer linear programming formulation with arc and precedence variables. We show that the underlying polytope shares some polyhedral properties with the ATSP polytope, which let us characterize large number of facets of our polytope. We convert these theoretical results into a branch-and-cut algorithm for the double TSP with two stacks. Our algorithm outperforms the existing exact methods and solves instances that were previously unsolved.

In this paper, we study a generalization of the Traveling Salesman Problem (*TSP*), namely the *double TSP with multiple stacks*. In this problem, n items have to be picked up in one city, stored in a vehicle having s identical stacks of finite capacity, and delivered to n customers in another city. We will assume that the pickup and the delivery cities are very far from each other, thus the pickup phase has to be entirely completed before the delivery phase starts. The pickup (resp. delivery) phase consists in performing a Hamiltonian circuit, *i.e.*, starting from a depot, the n pickup (resp. delivery) locations have to be visited in sequence exactly once before coming back to the depot. Each time a new item is picked up, it is stored on the top of an available stack of the vehicle according to its capacity and no rearrangement of the stacks is allowed. During the delivery circuit the stacks are unloaded following a last-in-first-out policy, that is, only items currently on the top of their stack can be delivered. The goal is to find the pickup and delivery circuits which minimize the total traveled distance, subject to the last-in-first-out consistency.

* Corresponding author. Tel.: +33 1 49 40 4071; fax: +33 1 48 26 0712.
E-mail address: roberto.wolfler@lipn.fr (R. Wolfler Calvo).

http://dx.doi.org/10.1016/j.disopt.2016.04.005
1572-5286/

The double TSP with multiple stacks is introduced in [1] as a fleet management project initiated in cooperation with a software company. The problem arises from real-world applications. As the authors point out in [1], the items to be transported are usually standardized Euro Pallets, which are identical from a packing point of view. Moreover, repacking is not allowed because of insurance issues.

The double TSP with multiple stacks is NP-hard since, when the vehicle has only one stack, it corresponds to the Asymmetric Traveling Salesman Problem (*ATSP*): indeed, in this case, due to the last-in-first-out policy, the delivery circuit is nothing but the pickup circuit performed in the reverse order. Moreover, deciding if a given pair of pickup and delivery circuits satisfies the last-in-first-out policy is NP-complete [2]. It becomes polynomial when the number of stacks is fixed [3] or if the stacks have infinite capacity [4,2].

Since its first appearance, the double TSP with multiple stacks has received increasing attention. Both exact algorithms and heuristics have been designed for this problem over the past few years. Regarding the exact algorithms, in [5,6], the authors design a procedure to iteratively generate the k-best ATSP pickup and delivery solutions and to find the best combination satisfying the last-in-first-out consistency. Several exponential and polynomial size mixed integer linear programming formulations have been proposed and tested in branch-and-cut frameworks [7,8]. An additive branch-and-bound algorithm [9] has been developed for the case with two stacks. In [10], the authors adapt a branch-and-cut algorithm for the pickup and delivery TSP with multiple stacks to the double TSP with multiple stacks.

From a computational point of view, these algorithms clearly show that the double TSP with multiple stacks is extremely hard to solve with exact methods. In particular, the difficulty of the problem increases with the capacity of the stacks [8]. As a consequence, given a number of items equal to the total capacity, the hardest case is the double TSP with two stacks. Currently, no algorithm efficiently solves instances with capacity greater than seven.

In this paper, we first focus on the double TSP with multiple stacks of infinite capacity. Section 1 is devoted to notation and definitions. In Section 2, we introduce an integer linear programming formulation with arc and precedence variables. We then show in Section 3 that the underlying polytope shares some polyhedral properties with the ATSP polytope. These links let us characterize a super-polynomial number of facets of our polytope. Afterwards, in Section 4, we strengthen our formulation by exploiting the last-in-first-out consistency of the pickup and delivery circuits. In Section 5, we convert these theoretical results into a branch-and-cut algorithm for the double TSP with two stacks. It turns out that our algorithm outperforms the existing exact methods and solves new instances of the benchmark from the literature—see Section 6.

1. Definitions

Given a set $S \subseteq \mathbb{R}^m$, $\text{conv}(S)$ is the convex hull of S; the symbol $\dim(S)$ denotes the dimension of the affine hull of S. Given $S \subseteq \mathbb{R}^n \times \mathbb{R}^d$, its projection into $\mathbb{R}^n$ is the set $\mathit{proj}_x(S) = \{x \in \mathbb{R}^n : \exists y \in \mathbb{R}^d \text{ such that } (x, y) \in S\}$. The projection $\mathit{proj}_y(S)$ of S into $\mathbb{R}^d$ is defined in an analogous manner. For S a finite set, $x \in \mathbb{R}^{|S|}$ and $H \subseteq S$, we write $x(H)$ for $\sum_{h\in H} x_h$.

We denote by G_n the complete digraph having $V = \{0, \dots, n\}$ as vertex set and $A = \{(i, j) : i \neq j \in V\}$ as arc set. A *circuit* of G_n is a set of arcs that induces a connected subgraph in which every vertex has exactly one entering and one leaving arc. Its *length* is the number of arcs it contains. A circuit is said *Hamiltonian* if its corresponding subgraph contains all the vertices. The *reverse* of a circuit C, denoted by $\overleftarrow{C}$, is the circuit composed of the opposite arcs of C.

A relation $\prec$ on $\{1, \dots, n\}$ is a *linear ordering* if it is reflexive, antisymmetric, transitive and total. Such a relation is represented by an order $v_1, \dots, v_n$ of $\{1, \dots, n\}$, where $v_i \prec v_j$ whenever $i < j$. It is noteworthy that a Hamiltonian circuit $C = \{(0, v_1), (v_1, v_2), \dots, (v_{n-1}, v_n), (v_n, 0)\}$ of G_n induces a linear ordering $v_1, \dots, v_n$ of $\{1, \dots, n\}$. This ordering will be denoted by $\prec_C$. Moreover, the converse holds because G_n is complete. Hence, such a Hamiltonian circuit C will also be written $C = 0, v_1, \dots, v_n, 0$.