

Load-Dependent and Precedence-Based Models for Pickup and Delivery Problems

Luis Gouveia^{a,1}, Mario Ruthmair^{b,c,2,*}

^a*ULisboa, Faculdade de Ciências, Dept de Estatística e Investigação Operacional, and Centro de Matemática, Aplicações Fundamentais e Investigação Operacional (CMAF+CIO), Lisbon, Portugal*

^b*Institute of Computer Graphics and Algorithms, Vienna University of Technology, Vienna, Austria*

^c*Mobility Department, Austrian Institute of Technology, Vienna, Austria*

Abstract

We address the one-to-one multi-commodity pickup and delivery traveling salesman problem (m -PDTSP) which is a generalization of the TSP and arises in several transportation and logistics applications. The objective is to find a minimum-cost directed Hamiltonian path which starts and ends at given depot nodes and such that the demand of each given commodity is transported from the associated source to its destination and the vehicle capacity is never exceeded. In contrast, the many-to-many one-commodity pickup and delivery traveling salesman problem (1-PDTSP), just considers a single commodity and each node can be a source or target for units of this commodity. We show that the m -PDTSP is equivalent to the 1-PDTSP with additional precedence constraints defined by the source-destination pairs for each commodity and explore several models based on this equivalence. In particular, we consider layered graph models for the capacity constraints and introduce new valid inequalities for the precedence relations. Especially for tightly capacitated instances with a large number of commodities our branch-and-cut algorithms outperform the existing approaches. For the uncapacitated m -PDTSP (which is known as the sequential ordering problem) we are able to solve to optimality several open instances from the TSPLIB and SOPLIB.

Keywords: Transportation, Traveling Salesman, Sequential ordering problem, Pickup and Delivery, Precedence constraints

1. Introduction

In this paper we propose a new approach for the one-to-one multi-commodity pickup and delivery traveling salesman problem (m -PDTSP) introduced

*Corresponding author

Email addresses: legouveia@fc.ul.pt (Luis Gouveia), mario.ruthmair@ait.ac.at (Mario Ruthmair)

¹This work is supported by National Funding from FCT - Fundação para a Ciência e a Tecnologia, under the project: PEst-OE/MAT/UI0152

²This work is supported by the Vienna Science and Technology Fund (WWTF) through project ICT10-027.

by Hernández-Pérez & Salazar-González (2009). The problem arises in several transportation and logistics applications. The m -PDTSP generalizes the well known travelling salesman problem (TSP) as well as two other variants which in turn also generalize the TSP. To better contextualize the m -PDTSP we will start by introducing briefly the other three variants, pointing out relations between the four problems as well as stating one of the main results of this paper.

We first consider the TSP (Lawler et al., 1985), or more precisely the asymmetric version since all the problems discussed here are defined in a directed graph $G = (V, A)$. For each arc $(i, j) \in A$, a travel distance (or cost) c_{ij} of going from i to j is given. The objective is to find a minimum cost Hamiltonian tour. Many formulations have been presented for this problem (see, for instance Roberti & Toth (2012), as probably the latest such reference) and we also refer the reader to the well known formulation by Dantzig et al. (1954) (DFJ) that will be stated in Section 3 as a subformulation for all the formulations presented and discussed in this paper.

The first generalization we consider is the precedence constrained TSP (PC-TSP) where a set K of pairs of nodes $(s_k, d_k), \forall k \in K$, is given as an input of the problem. In this variant we consider a special node, node 0 as a depot and where the tour starts and ends. As before, the objective is to find a minimum cost Hamiltonian circuit, but now we have the additional constraint that for each $k \in K$, node s_k must precede node d_k in the tour. We refer the reader to the papers by Balas et al. (1995), Ascheuer et al. (2000), and Gouveia & Pesneau (2006). Cut-like inequalities specific for the precedence case and generalizing the well known cut inequalities that arise in the DFJ formulation, have been proposed in the first paper. These sets of constraints will be referred to in Section 4. We also observe that this problem is often defined as searching for a minimum cost Hamiltonian path between a source node 0 and destination node $n + 1$. The two variants are obviously equivalent. Also, the PCTSP is known as the sequential ordering problem (SOP). From now on, we will keep the Hamiltonian path alternative for describing the subsequent variants including the problem studied in this paper.

A second variant of the TSP is the so-called many-to-many one-commodity pickup and delivery traveling salesman problem (1-PDTSP) and has been introduced by Hernández-Pérez & Salazar-González (2003). In this problem and as stated before, we consider a node set V with a start and end depot 0 and $n + 1$, respectively, and the set of customers $V_c = \{1, \dots, n\}$. We also consider a vehicle of capacity Q and a single commodity, and each node can be a source or target for units of this commodity. Values $\rho_j, \forall j \in V$, represent the customer demands: Nodes with $\rho_j > 0$ and $\rho_j < 0$ are denoted pickup and delivery customers, respectively. Nodes with $\rho_j = 0$ also need to be visited without changing the vehicle load. Again, we want to find a Hamiltonian path from 0 to $n + 1$ satisfying all customer demands and the given vehicle capacity Q . It is \mathcal{NP} -hard to find a feasible solution for the 1-PDTSP as shown by Hernández-Pérez & Salazar-González (2003). The papers by Hernández-Pérez & Salazar-González (2004, 2007) present several models and valid inequalities for the 1-PDTSP and branch-and-cut algorithms to solve it. One of these models will be reviewed in Section 4. Clearly, if one ignores the vehicle capacity, the 1-PDTSP reduces to the TSP.

So far, we have described two variants that generalize the TSP. As mentioned before, in this paper we study a new approach for the m -PDTSP. This problem

can also be viewed as a generalization of the SOP and the 1-PDTSP.

In the m -PDTSP there are m commodities $K = \{1, \dots, m\}$, each $k \in K$ associated with a demand q_k , a source $s_k \in V \setminus \{n+1\}$, and a destination $d_k \in V \setminus \{0\}$. We assume $s_k \neq d_k$ and $q_k > 0$. A customer j can be the source of several commodities and the destination of other commodities. As in the 1-PDTSP we also consider a vehicle capacity $Q > 0$. We assume that $q_k \leq Q$ for all $k \in K$. The objective is to find a minimum cost Hamiltonian path between nodes 0 and $n+1$, such that i) for each commodity $k \in K$ source s_k is visited before destination d_k , ii) q_k units are transported from s_k to d_k , and iii) the vehicle capacity is an upper bound of the vehicle load for each arc on the path from 0 to $n+1$.

As pointed out by Hernández-Pérez & Salazar-González (2009) the m -PDTSP generalizes the 1-PDTSP. We simply aggregate the different flows into a single one. The customer demands of the equivalent 1-PDTSP are defined by the load changes when the vehicle visits a customer in the m -PDTSP. Again, and as also pointed out in Hernández-Pérez & Salazar-González (2009), if one ignores the vehicle capacity in the m -PDTSP, one obtains the SOP since the precedence between source and destination for each commodity must be maintained.

The m -PDTSP is \mathcal{NP} -hard since it generalizes all the variants described here which are also known to be \mathcal{NP} -hard. Hernández-Pérez & Salazar-González (2009) present two solution approaches, both based on Benders decomposition of a path and a multi-commodity flow model, respectively. The multi-commodity flow model will be revisited in Section 3. Their branch-and-cut algorithms are based on models in the natural variable space, i.e., only use binary variables for arcs A . These approaches usually achieve excellent results in terms of solution runtime for loosely-constrained problem instances, i.e., when only a few commodities have to be considered or the given vehicle capacity is large in relation to the demands. In these cases only a few violated inequalities have to be added within the cutting plane phase. Additionally, the reduced size of the initial model makes it possible to quickly solve the corresponding linear programming (LP) relaxation. However, when considering problem instances with many commodities and/or a tight vehicle capacity several weaknesses of these approaches show up, namely that the basic model provides only a quite weak LP relaxation value leading to a large number of branch-and-bound nodes and making it necessary to add many violated inequalities. Rodríguez-Martín & Salazar-González (2012) also propose several heuristic approaches for the m -PDTSP to obtain high-quality solutions for larger instances for which exact approaches cannot obtain satisfying results within reasonable time. They present a simple nearest neighbor heuristic to construct a solution followed by an improvement phase based on 2-opt, 3-opt, and restricted mixed integer programming neighborhood structures. We conclude this literature review by pointing to the overview on further pickup and delivery problems given in Berbeglia et al. (2007).

The models in this paper are mostly based on a new result stating that the m -PDTSP is equivalent to the 1-PDTSP with additional precedence constraints defined by the origin-destination pairs for each commodity. That is, in a loose sense the m -PDTSP combines together the two previous variants. The advantage of using this relation to model the m -PDTSP is that we are able to model the capacity constraints just by considering a single commodity and this helps considerably in running times. The precedence relations are ensured separately by adding valid inequalities from the SOP, see Balas et al. (1995) and Ascheuer

et al. (2000). We also introduce new inequalities based on sequences and logical implications of precedence relations which are able to further close the LP gaps, especially for instances with a large number of precedence constraints.

Also, we present alternative ways to model the capacity constraints based on load-dependent layered graphs which improve the LP bounds for tight capacities. In particular we consider a formulation based on a 3-dimensional layered graph that combines position and load together and leads to tighter bounds at the cost of a larger sized model.

Our branch-and-cut algorithm to solve the m -PDTSP consists of several preprocessing methods, primal heuristics, and separation routines for the SOP inequalities. Especially for tightly capacitated instances with a large number of commodities we are able to outperform the approaches by Hernández-Pérez & Salazar-González (2009). In our experiments, we also consider the uncapacitated variant of the m -PDTSP, i.e., the SOP. Here, an adapted variant of our branch-and-cut algorithm is able to solve to optimality several open instances from the TSPLIB and SOPLIB.

The remainder of the article is structured as follows: In Section 2 we present reduction and preprocessing techniques for the m -PDTSP, Section 3 revises existing models, Section 4 discusses the transformation to a single-commodity problem, Section 5 introduces layered graph models for the capacity constraints, Section 6 describes our branch-and-cut algorithms, Section 7 shows experimental results, and Section 8 concludes the paper.

2. Preprocessing

In this section we discuss some problem reductions and relevant problem properties which will be used to reduce and strengthen the models discussed in this paper. Additionally, these tests and properties may lead to an early detection of infeasibility of an instance.

2.1. Commodities

A commodity $k \in K$ is called transitive if there exist commodities $k_1, k_2 \in K \setminus \{k\}$ with $s_{k_1} = s_k, d_{k_1} = s_{k_2}, d_{k_2} = d_k$. It can be easily seen that the set of feasible solutions is not modified if a transitive commodity is removed from set K and the demands of the corresponding commodities k_1 and k_2 are appropriately modified, i.e., $q'_{k_1} = q_{k_1} + q_k$ and $q'_{k_2} = q_{k_2} + q_k$. We perform this reduction step for all transitive commodities.

2.2. Precedence Relations

The source-destination pairs $(s_k, d_k), \forall k \in K$, induce an acyclic precedence graph $P = (V, R)$ with R being the transitive closure of $R' = \{(s_k, d_k) : k \in K\} \cup \{(0, i) : i \in V \setminus \{0\}\} \cup \{(i, n+1) : i \in V \setminus \{n+1\}\}$. Clearly, arc $(j, i) \in A$ can be removed from the original graph G if $(i, j) \in R$ since it cannot appear in any feasible solution. Additionally, arc $(i, j) \in A$ can be removed if $(i, j) \in R$ is transitive, i.e., for some $k \in V, (i, k), (k, j) \in R$ (cf. Balas et al., 1995). Let $\bar{R} \subseteq R$ be the subset of non-transitive precedence relations.

2.3. Vehicle Load Bounds

For each node $j \in V$ we define net demands $\rho_j := \sum_{k:j=s_k} q_k - \sum_{k:j=d_k} q_k$, representing the load change of the vehicle when visiting node $j \in V$. For each arc $(i, j) \in A$ we compute lower and upper bounds l_{ij} and u_{ij} on the vehicle load, respectively. The load on arcs going out of and coming in to the depot is fixed and defined by the commodities starting or ending in the depot, i.e., $l_{0i} = u_{0i} = \sum_{k \in K, s_k=0} q_k, \forall (0, i) \in A$, and $l_{i, n+1} = u_{i, n+1} = \sum_{k \in K, d_k=n+1} q_k, \forall (i, n+1) \in A$. This situation is different from the 1-PDTSP where the initial vehicle load cannot be derived a priori since it depends on the visiting sequence. To calculate the load bounds for all other arcs $(i, j) \in A, i \neq 0, j \neq n+1$, we use some ideas from Hernández-Pérez & Salazar-González (2009) and extend them in the following way. For each commodity $k \in K$ we define the set of nodes $V_k^{\text{in}} \subseteq V$ which have to be on the path from s_k to d_k in any feasible solution. Set $V_k^{\text{out}} \subset V$ includes nodes which cannot be on the path from s_k to d_k in any feasible solution:

$$\begin{aligned} V_k^{\text{in}} &:= \{i \in V : i = s_k \vee i = d_k \vee (s_k, i), (i, d_k) \in R\} \\ V_k^{\text{out}} &:= \{i \in V : (i, s_k) \in R \vee (d_k, i) \in R\} \end{aligned}$$

Similarly, we define set A_k^{in} consisting of arcs (i, j) which – if used in a solution – have to be on the path from s_k to d_k . Set A_k^{out} includes arcs (i, j) which – if used in a solution – cannot be on the path from s_k to d_k :

$$\begin{aligned} A_k^{\text{in}} &:= \{(i, j) \in A : i \in V_k^{\text{in}} \setminus \{d_k\} \vee j \in V_k^{\text{in}} \setminus \{s_k\} \vee (s_k, i), (j, d_k) \in R\} \\ A_k^{\text{out}} &:= \{(i, j) \in A : i = d_k \vee j = s_k \vee i \in V_k^{\text{out}} \vee j \in V_k^{\text{out}}\} \end{aligned}$$

Then, lower and upper load bounds for the flows in the arcs can be defined as follows:

$$l_{ij} = \sum_{k:(i,j) \in A_k^{\text{in}}} q_k, \quad u_{ij} = \min\{Q - \max\{0, -\rho_i, \rho_j\}, \sum_{k:(i,j) \notin A_k^{\text{out}}} q_k\}$$

To further strengthen the load bounds we consider all feasible paths P_{hijk} of length three and update the bounds in the following way:

$$l_{ij} = \min_{P_{hijk}} \max\{l_{hi} + \rho_i, l_{ij}, l_{jk} - \rho_j\}, \quad u_{ij} = \max_{P_{hijk}} \min\{u_{hi} + \rho_i, u_{ij}, u_{jk} - \rho_j\}$$

These bounds are used for tightening the models presented in this article.

Furthermore, for each arc $(i, j) \in A$ we define sets A_{ij}^- and A_{ij}^+ of all feasible preceding and succeeding arcs, respectively:

$$\begin{aligned} A_{ij}^- &:= \{(k, i) \in A : k \neq j, (j, k) \notin R, (k, l) \notin R \text{ for some } l \neq i \text{ with } (l, j) \in R\} \\ A_{ij}^+ &:= \{(j, k) \in A : k \neq i, (k, i) \notin R, (l, k) \notin R \text{ for some } l \neq j \text{ with } (i, l) \in R\} \end{aligned}$$

Then, for each arc $(i, j) \in A, i \neq 0, j \neq n+1$, we define lower and upper bounds l_{ij}^- and u_{ij}^- , respectively, on the vehicle load coming into node i and bounds l_{ij}^+ and u_{ij}^+ on the load going out of node j , assuming that arc (i, j) is traversed, as follows:

$$\begin{aligned} l_{ij}^- &= \sum_{k:A_{ij}^- \subseteq A_k^{\text{in}} \vee (i \neq s_k \wedge (i, j) \in A_k^{\text{in}})} q_k, & u_{ij}^- &= \min\{u_{ij} - \rho_i, \max_{(k,i) \in A} u_{ki}\} \\ l_{ij}^+ &= \sum_{k:A_{ij}^+ \subseteq A_k^{\text{in}} \vee (j \neq d_k \wedge (i, j) \in A_k^{\text{in}})} q_k, & u_{ij}^+ &= \min\{u_{ij} + \rho_j, \max_{(j,k) \in A} u_{jk}\} \end{aligned}$$

Arcs $(i, j) \in A$ can be removed if $l_{ij} > u_{ij}$ or $l_{ij}^- > u_{ij}^-$ or $l_{ij}^+ > u_{ij}^+$. These preprocessing steps may help to decide in an early stage of the solution process whether a particular problem instance is infeasible or unconstrained with respect to a given vehicle capacity Q .

3. Multi-Commodity Flow Model for m -PDTSP (Hernández-Pérez & Salazar-González, 2009)

Before describing the models we introduce some notation: The set of arcs going out of some set $S \subset V$ is denoted by $\delta^+(S) := \{(i, j) \in A : i \in S, j \notin S\}$. Similarly, we use $\delta^-(S) := \{(i, j) \in A : i \notin S, j \in S\}$ for the set of arcs coming into set S . If $S = \{i\}$ we simply write $\delta^+(i)$ and $\delta^-(i)$, respectively. Furthermore, for a set of arcs $A' \subseteq A$ we write $v(A') := \sum_{(i,j) \in A'} v_{ij}$ to denote the sum of variables v associated to arcs A' . We write $v(S) := \sum_{(i,j) \in A, i,j \in S} v_{ij}$ for the sum of variables of arcs within node set $S \subseteq V$. Similarly, we write $v(S, S') := \sum_{(i,j) \in A, i \in S, j \in S'} v_{ij}$ for the sum of variables of arcs going from set $S \subseteq V$ to set $S' \subseteq V$. M_L denotes the LP relaxation of model M . $F(M)$ denotes the set of feasible solutions of model M . $Proj_v(S)$ denotes the projection of set S into the space defined by variables v .

Since the feasible solutions for the problem under study are Hamiltonian paths from 0 to $n + 1$, we consider the following generic model for the problem. We use binary variables $x_{ij}, \forall (i, j) \in A$:

$$\begin{aligned} \min \quad & \sum_{(i,j) \in A} c_{ij} x_{ij} & (1) \\ \text{s.t.} \quad & x(\delta^+(i)) = 1 & \forall i \in V \setminus \{n+1\} & (2) \\ & x(\delta^-(i)) = 1 & \forall i \in V \setminus \{0\} & (3) \\ & x(\delta^+(S)) \geq 1 & \forall S \subseteq V \setminus \{n+1\} & (4) \\ & \{(i, j) : x_{ij} = 1\} & \text{supports flows for each } k \in K & (5) \\ & & \text{and satisfies vehicle capacity} \\ & x_{ij} \in \{0, 1\} & \forall (i, j) \in A & (6) \end{aligned}$$

In some of the models presented next we will provide alternative ways of modeling the connectivity constraints (4). These situations will be indicated later on but for simplicity we present the generic model with (4) which are the most well known constraints for guaranteeing connectivity. The system (1)–(4) and (6) is the well known DFJ model mentioned in Section 1. These constraints are also used in models for the 1-PDTSP and m -PDTSP in previous papers (e.g., Hernández-Pérez & Salazar-González, 2004, 2007, 2009). Note that constraints (4), although exponential in number, can be easily implicitly included in the model by a cutting plane approach based on finding violated inequalities with max-flow computations (see, e.g., Padberg & Rinaldi, 1991).

The flow model by Hernández-Pérez & Salazar-González (2009) is based on the generic scheme mentioned before. Flows and the vehicle capacity in constraints (5) are ensured by adding for each commodity $k \in K$ and each arc $(i, j) \in A$, the flow variable f_{ij}^k indicating the flow on arc (i, j) of commodity k

as well as the following set of flow conservation and capacity constraints:

$$f^k(\delta^+(i)) - f^k(\delta^-(i)) = \begin{cases} q_k & \text{if } i = s_k \\ -q_k & \text{if } i = d_k \\ 0 & \text{else} \end{cases} \quad \forall i \in V \setminus V_k^{\text{out}}, \forall k \in K \quad (7)$$

$$\sum_{k \in K} f_{ij}^k \leq Qx_{ij} \quad \forall (i, j) \in A \quad (8)$$

$$f_{ij}^k \geq 0 \quad \forall (i, j) \in A, \forall k \in K \quad (9)$$

Note that we have reduced the size of the model by eliminating the flow conservation constraints for all nodes in V_k^{out} for each commodity k . Also and as mentioned by Hernández-Pérez & Salazar-González (2009) the LP relaxation of the model can be improved by replacing constraints (8)–(9) with the following well known modeling strengthening constraints of multi-commodity flow models extended by information obtained in preprocessing:

$$l_{ij}x_{ij} \leq \sum_{k \in K} f_{ij}^k \leq u_{ij}x_{ij} \quad \forall (i, j) \in A \quad (10)$$

$$0 \leq f_{ij}^k \begin{cases} = 0 & \text{if } (i, j) \in A_k^{\text{out}} \\ = q_k x_{ij} & \text{if } (i, j) \in A_k^{\text{in}} \\ \leq q_k x_{ij} & \text{else} \end{cases} \quad \forall (i, j) \in A, \forall k \in K \quad (11)$$

We denote by **MCF** the generic model (1)–(4), (6), with the multi-commodity flow system (7), (10)–(11).

4. Relating the m -PDTSP to the 1-PDTSP (with Precedence Constraints)

In this section we suggest new models for the m -PDTSP that are motivated by observing that the m -PDTSP is equivalent to the 1-PDTSP with additional precedence constraints defined by the origin-destination pairs (s_k, d_k) for each commodity $k \in K$. As far as we know, and as pointed out in the introduction this “equivalence” relation has been neither stated nor used before. Weaker related relations have been pointed out by Hernández-Pérez & Salazar-González (2009) stating that the two problems: i) the 1-PDTSP using net demands ρ (without considering any precedence relations) and ii) the TSP with precedence constraints defined by the commodities (with unlimited vehicle capacity) are relaxations of the m -PDTSP. Essentially, we are saying that by adequately combining these two relaxed problems we obtain a problem equivalent to the m -PDTSP.

To motivate the relation between the m -PDTSP and the 1-PDTSP with precedence constraints, we show next how to transform the MCF model described in the previous section into a different and equivalent model where this relation is enhanced.

4.1. Introducing Scaled Flow Variables

We introduce scaled flow variables g_{ij}^k and use equalities

$$f_{ij}^k = q_k g_{ij}^k \quad \forall (i, j) \in A, \forall k \in K, \quad (12)$$

to rewrite (7) and (11) as follows:

$$g^k(\delta^+(i)) - g^k(\delta^-(i)) = \begin{cases} 1 & \text{if } i = s_k \\ -1 & \text{if } i = d_k \\ 0 & \text{else} \end{cases} \quad \forall i \in V \setminus V_k^{\text{out}}, \forall k \in K \quad (13)$$

$$0 \leq g_{ij}^k \begin{cases} = 0 & \text{if } (i, j) \in A_k^{\text{out}} \\ = x_{ij} & \text{if } (i, j) \in A_k^{\text{in}} \\ \leq x_{ij} & \text{else} \end{cases} \quad \forall (i, j) \in A, \forall k \in K \quad (14)$$

4.2. Aggregating the Flows

Next, we sum up equalities (7) for all commodities $k \in K$ and obtain:

$$\sum_{k \in K} f^k(\delta^+(i)) - \sum_{k \in K} f^k(\delta^-(i)) = \sum_{k: i=s_k} q_k - \sum_{k: i=d_k} q_k = \rho_i \quad \forall i \in V \quad (15)$$

By using aggregated flow variables $f_{ij}, \forall (i, j) \in A$, and equalities

$$f_{ij} = \sum_{k \in K} f_{ij}^k \quad \forall (i, j) \in A, \quad (16)$$

we can rewrite the aggregated flow conservation constraints (15) and the capacity constraints (10) as the following single-commodity flow (SCF) system:

$$f(\delta^+(i)) - f(\delta^-(i)) = \rho_i \quad \forall i \in V \quad (17)$$

$$l_{ij}x_{ij} \leq f_{ij} \leq u_{ij}x_{ij} \quad \forall (i, j) \in A \quad (18)$$

4.3. Combining the Scaled Flow System with the Aggregated Flow System

We denote by ‘‘Transformed MCF’’ (**TMCF**) the model MCF with the flow system (7), (10)–(11) on f_{ij}^k variables replaced by the scaled flow system (13)–(14) on g_{ij}^k variables, the aggregated SCF system (17)–(18), and the linking constraints

$$f_{ij} = \sum_{k \in K} q_k g_{ij}^k \quad \forall (i, j) \in A. \quad (19)$$

Table 1 shows the complete model TMCF. It is now easy to see that in terms of integer solutions, the model TMCF is equivalent to the model MCF. One direction has already been proved with the given transformation. To see the reverse situation, note that from a given solution feasible for the model TMCF, we obtain a feasible solution for the model MCF simply by setting the f_{ij}^k variables as defined by linking constraints (12).

Thus, we have just proved that:

Result 4.1. *Under the transformation (12) the model TMCF is equivalent to the model MCF.*

Since a similar equivalence holds with respect to the corresponding LP relaxations (where we replace (6) with $0 \leq x_{ij} \leq 1, \forall (i, j) \in A$), we can conclude that the two models provide the same LP bound.

Table 1: Model TMCF

$\min \sum_{(i,j) \in A} c_{ij} x_{ij}$	(1)
$x(\delta^+(i)) = 1$	$\forall i \in V \setminus \{n+1\}$ (2)
$x(\delta^-(i)) = 1$	$\forall i \in V \setminus \{0\}$ (3)
$x(\delta^+(S)) \geq 1$	$\forall S \subseteq V \setminus \{n+1\}$ (4)
$x_{ij} \in \{0, 1\}$	$\forall (i, j) \in A$ (6)
$g^k(\delta^+(i)) - g^k(\delta^-(i)) = \begin{cases} 1 & \text{if } i = s_k \\ -1 & \text{if } i = d_k \\ 0 & \text{else} \end{cases}$	$\forall i \in V \setminus V_k^{\text{out}}, \forall k \in K$ (13)
$0 \leq g_{ij}^k \begin{cases} = 0 & \text{if } (i, j) \in A_k^{\text{out}} \\ = x_{ij} & \text{if } (i, j) \in A_k^{\text{in}} \\ \leq x_{ij} & \text{else} \end{cases}$	$\forall (i, j) \in A, \forall k \in K$ (14)
$f(\delta^+(i)) - f(\delta^-(i)) = \rho_i$	$\forall i \in V$ (17)
$l_{ij} x_{ij} \leq f_{ij} \leq u_{ij} x_{ij}$	$\forall (i, j) \in A$ (18)
$f_{ij} = \sum_{k \in K} q_k g_{ij}^k$	$\forall (i, j) \in A$ (19)

4.4. Relating the m -PDTSP with the 1-PDTSP with Precedence Constraints

Consider the model that is obtained from TMCF by removing the linking constraints (19). We denote by Weak TMCF (**WTMCF**) the model obtained in this way. We show next that the model WTMCF is still a valid model for the problem, although, obviously, with a weaker LP relaxation.

Theorem 4.2. *Model WTMCF is a valid formulation for the m -PDTSP.*

Proof. We show this by induction on the number of commodities K .

$m = 1$: In case of a single commodity net demand values are set to $\rho_i = 0, \forall i \in V \setminus \{s_1, d_1\}$, and $\rho_{s_1} = q_1$ and $\rho_{d_1} = -q_1$. Here, we do not even need to explicitly ensure that s_1 is visited before d_1 since the SCF system (17)–(18) already forbids to visit d_1 before s_1 because of the negative value ρ_{d_1} and the lower vehicle load bound 0. The consequence is that the m -PDTSP with $K = \{1\}$ is equivalent to the 1-PDTSP which can be modeled by the generic part (1)–(4), and (6), and flow system (17)–(18) (see Hernández-Pérez & Salazar-González, 2004).

Inductive step: We assume that model WTMCF is valid for the $(m - 1)$ -PDTSP with commodities $K = \{1, \dots, m - 1\}$. We want to show that model WTMCF stays valid when adding a further commodity m . The additional flow system (13)–(14), for $k = m$ ensures that s_m is visited before d_m . Furthermore, we observe that exactly two net demand values change, i.e., $\rho'_{s_m} = \rho_{s_m} + q_m$ and $\rho'_{d_m} = \rho_{d_m} - q_m$. The SCF inequalities (17)–(18) for nodes $i = s_m, d_m$ ensure that the additional demand q_m is considered with respect to the vehicle load bounds. \square

We observe that the aggregated flow system on variables f_{ij} (see third box in Table 1) guarantees that the capacity constraints and the net demands are satisfied and corresponds to the flow system in formulations for the 1-PDTSP (e.g., Hernández-Pérez & Salazar-González, 2004, 2007). Also, the g_{ij}^k system (see second box in Table 1) guarantees the precedence relations for each commodity $k \in K$.

This decomposition puts in evidence the fact that we can model the m -PDTSP as the 1-PDTSP model together with any set of precedence constraints guaranteeing the precedence relations defined by the commodity pairs. In the model WTMCF, these precedence constraints are modelled with the flow system (13)–(14) on g_{ij}^k variables. In the next subsection we will describe a different alternative.

4.5. Modeling the Precedence Constraints with SOP Inequalities

According to Balas et al. (1995) we consider cut-like inequalities known from the SOP, i.e., the simple (π, σ) -inequalities which are described as follows: For each commodity $k \in K$ we define a set of relevant nodes $V^k = V \setminus V_k^{\text{out}}$ and the corresponding inequalities are defined as follows:

$$x(S, V^k \setminus S) \geq 1 \quad \forall S \subset V^k, s_k \in S, d_k \in V^k \setminus S, \forall k \in K \quad (20)$$

Similar to connection cuts (4), these inequalities associated to one particular commodity k ensure a path from s_k to d_k in a reduced graph excluding all nodes which have to be visited before s_k or after d_k (including nodes 0 and $n+1$). Inequalities (20) can be separated in polynomial time for each commodity $k \in K$ by max-flow computations in a similar way as the connection cuts (4) but in a support graph induced by node set V^k .

We denote by **CUTK**, the model WTMCF with the flow system (13)–(14) replaced by SOP cuts (20), i.e., objective (1), degree and connectivity constraints (2)–(4), integrality constraints (6), the SCF system (17)–(18), and SOP cuts (20). As a straightforward consequence of the max-flow min-cut theorem (Ahuja et al., 1993) we can state that the projection of the set of feasible solutions defined by the flow system (13)–(14) and $0 \leq x_{ij} \leq 1, \forall (i, j) \in A$, into the space of the x_{ij} variables is defined by the SOP cuts (20) and $0 \leq x_{ij} \leq 1$, that is:

Result 4.3. $Proj_x(F(WTMCF_L)) = Proj_x(F(CUTK_L))$.

This result states that the bounds obtained from the LP relaxations of model CUTK and WTMCF are the same. As pointed out in Section 4.4, the model just obtained produces an LP bound that is weaker than the LP bound produced by TMCF (since we lose the connection between the two sets of flow variables). The difference in LP bound is more evident for cases with tight capacity. Nevertheless, for instances with many commodities, models CUTK and/or WTMCF may be preferable to model TMCF (which is confirmed by our computational results), since they have far fewer variables.

4.6. Valid Inequalities

We can strengthen all the models discussed above by adding other families of precedence related cut-like inequalities to the model. Besides using the source-target pairs (s_k, d_k) for each commodity $k \in K$ to define associated SOP inequalities, we consider additional node pairs corresponding to non-transitive precedence relations starting in depot node 0 or ending in node $n+1$. To define all of the relevant pairs we consider

$$\begin{aligned} \tilde{R} = & \{(s_k, d_k) : k \in K\} \\ & \cup \{(0, i) : i \in V \setminus \{0, n+1\}, (j, i) \notin R, \forall j \in V \setminus \{0\}\} \\ & \cup \{(i, n+1) : i \in V \setminus \{0, n+1\}, (i, j) \notin R, \forall j \in V \setminus \{n+1\}\}. \end{aligned}$$

Similar to the inequalities (20) for each precedence relation $(i, j) \in \tilde{R}$ we define a set of relevant nodes $V^{ij} = V \setminus \{k : (k, i) \in R \vee (j, k) \in R\}$. Then, the corresponding SOP inequalities are given as follows:

$$x(S, V^{ij} \setminus S) \geq 1 \quad \forall S \subset V^{ij}, i \in S, j \in V^{ij} \setminus S, \forall (i, j) \in \tilde{R} \quad (21)$$

If $i = 0$, inequalities (21) are known as weak σ -inequalities, if $j = n + 1$ as weak π -inequalities, and if $i \neq 0$ and $j \neq n + 1$ we obtain the simple (π, σ) -inequalities (Balas et al., 1995), as already mentioned in Subsection 4.5. It is easy to see that these inequalities dominate connection cuts (4) due to the inclusion of the additional node pairs in \tilde{R} . We denote model CUTK with inequalities (20) replaced by (21) by **CUTR**, i.e., objective (1), degree constraints (2)–(3), integrality constraints (6), the SCF system (17)–(18), and SOP cuts (21). Note that the LP bound obtained from model CUTR is at least as good as the one from model CUTK and our experimental results indicate that for many instances it is clearly better. Additionally, for any violated SOP cut (21) we check if it can be lifted to a stronger π -, σ -, or (π, σ) -inequality: Depending on set S we may exclude further cut arcs on the left-hand side of (21). For details we refer to Balas et al. (1995).

Balas et al. (1995) have also proposed a different set of inequalities, the so called precedence cycle breaking constraints (PCB). We consider a subset of these constraints with $S \subset V \setminus \{n + 1\}$ and $i_1, i_3 \in S, i_2 \notin S$, with $(i_1, i_2), (i_2, i_3) \in R$. Then,

$$x(S, V \setminus S) \geq 2. \quad (22)$$

Essentially, what these inequalities say is that we need to cross the cut from S to $V \setminus S$ at least twice, once when going from i_1 to i_2 , and again in the subpath from node i_3 to $n + 1$. We generalize the concept motivating inequalities (22) by considering sequences of precedence relations $(i_1, i_2), \dots, (i_{k-1}, i_k) \in R, i_1, \dots, i_k \in V \setminus \{n + 1\}$, for odd values of $k \geq 3$. We require all odd indexed nodes to be in set $S \subset V \setminus \{n + 1\}$ and all even indexed nodes to be in set $V \setminus S$, i.e., $\{i_h : h \leq k, h \text{ odd}\} \subseteq S$ and $\{i_h : h \leq k, h \text{ even}\} \subseteq V \setminus S$. Due to this node assignment we have to cross the cut $(S, V \setminus S)$ at least $\lceil k/2 \rceil$ times to ensure a path from i_1 to $n + 1$. Thus, the corresponding inequality is defined as

$$x(S, V \setminus S) \geq \lceil k/2 \rceil. \quad (23)$$

Note that due to the rounded right-hand sided value, the inequalities (23) for sequences with k even are dominated by inequalities (23) for the same sequences without the last node. Note also that sequences including transitive precedence relations are dominated by the ones consisting only of non-transitive relations, as shown by Balas et al. (1995) for the PCB inequalities. To find non-dominated sequences we use transitive relations $(i, j) \in R \setminus \tilde{R}$ with $i, j \in V \setminus \{n + 1\}$, and search for the longest path $(i = i_1, i_2, \dots, i_k = j)$ in the precedence graph P . Note that all precedence relations along this path are non-transitive since otherwise there would be a longer path in P . If k is even we do not consider the corresponding pair $(i, j) \in R \setminus \tilde{R}$ for inequalities (23) for the reasons explained above.

Inequalities (22) and (23) can be separated in polynomial time by computing the max-flow in a support graph $G' = (V', A')$ with $V' = V \cup \{s, t\}$ extending

set V by artificial source and target nodes, and defining $A' = A \cup \{(s, i_h) : h \leq k, h \text{ odd}\} \cup \{(i_h, t) : h \leq k, h \text{ even}\}$, connecting the source and target nodes to the nodes fixed to S and $V \setminus S$, respectively. The capacities on the arcs incident to node s and t are set to 1. It is straightforward to see that the minimum cut $(S, V \setminus S)$ obtained from the max-flow from s to t is such that the nodes i_h for all $h = 1, \dots, k$, are assigned to the sets as defined above.

Since inequalities (22) and (23) consider the path starting from i_1 , passing through all nodes $i_h, h = 2, \dots, k$, and ending in an arbitrary node in $V \setminus S$, we can lift these cuts by excluding all nodes which have to be before i_1 or after the successor of i_k , i.e., $S' = \{j : (j, i_1) \in R \vee (i_k, j) \in R \setminus \tilde{R}\}$. Thus, we obtain the lifted inequalities

$$x(S \setminus S', V \setminus (S \cup S')) \geq \lceil k/2 \rceil. \quad (24)$$

Similar to the SOP cuts (23), inequalities (24) can be separated in polynomial time in support graph $G'' = (V' \setminus S', A' \setminus \{(i, j) : i \in S' \vee j \in S'\})$.

The sets of inequalities to be presented next are based on logical implications to fix variables in a branch-and-bound node (Ascheuer et al., 2000). We adopt the notation from Balas et al. (1995) and write $\pi(j) := \{i : (i, j) \in R\}$ for the set of predecessors for node $j \in V$. Similarly, we use $\sigma(i) := \{j : (i, j) \in R\}$ to denote the corresponding set of successors. If $x_{ij} = 1$ for an arc $(i, j) \in A, i, j \in V_c$, then other (non-trivial) arcs can be fixed to zero, e.g., $x(\pi(i), \sigma(j)) = x(\sigma(j), \pi(i)) = x(\sigma(i), \pi(j)) = x(\pi(j), \sigma(i)) = 0$. We create valid inequalities based on these implications:

$$x(\{i, j\}) + x(\{k, l\}) \leq 1 \quad \forall i, j \in V_c, i \neq j, \forall k \in \pi(i), \forall l \in \sigma(j) \quad (25)$$

$$x(\{i, j\}) + x(k, \sigma(j)) \leq 1 \quad \forall i, j \in V_c, i \neq j, \forall k \in \pi(i) \quad (26)$$

$$x(\{i, j\}) + x(\sigma(j), k) \leq 1 \quad \forall i, j \in V_c, i \neq j, \forall k \in \pi(i) \quad (27)$$

$$x(\{i, j\}) + x(\pi(i), l) \leq 1 \quad \forall i, j \in V_c, i \neq j, \forall l \in \sigma(j) \quad (28)$$

$$x(\{i, j\}) + x(l, \pi(i)) \leq 1 \quad \forall i, j \in V_c, i \neq j, \forall l \in \sigma(j) \quad (29)$$

The validity of these inequalities can be easily shown by using node degree constraints (2) and (3). We add violated inequalities (25)–(29) within a cutting plane algorithm by examining them one by one.

Finally, in case of a violated inequality (4), (22), and (23) for some set S we check if the corresponding rounded capacity cut (Letchford & Salazar-González, 2006) in the context of the m -PDTSP is stronger:

$$x(\delta^+(S)) \geq \left\lceil \frac{\sum_{k: s_k \in S \wedge d_k \notin S} q_k}{\max_{(i, j) \in \delta^+(S)} u_{ij}} \right\rceil \quad (30)$$

Note that we can strengthen the right side by using the largest upper load bound over all cut arcs instead of the vehicle capacity.

5. Layered Graph Models

Models on layered graphs have been shown to provide strong LP bounds and lead to optimal solutions with short runtimes for several classes of problems, e.g., for tree problems (Gouveia et al., 2011, 2014b,a; Ruthmair & Raidl, 2011), TSP variants (Godinho et al., 2011, 2014; Abeledo et al., 2013), and location

problems (Ljubić & Gollowitzer, 2013). In these formulations paths are modeled in an expanded layered graph where the layers correspond to the position or time within the path. Since the layered graphs are acyclic, subtours are eliminated implicitly by the structure of this graph.

5.1. The Picard and Queyranne Formulation for the Capacity Constraints

In this subsection we show that the model by Picard & Queyranne (1978) (PQ) can be easily readapted to model the capacity constraints of the aggregated SCF system (17) and (18). We consider the variables z_{ij}^l for each arc $(i, j) \in A$ and each possible vehicle load $l \in L_{ij} := \{l_{ij}, \dots, u_{ij}\}$. Let $L_i := \bigcup_{(i,j) \in A} L_{ij}$ be the set of possible vehicle loads when leaving node $i \in V$. The load-dependent PQ system is defined as follows:

$$z^{l-\rho_j}(\delta^-(j)) = z^l(\delta^+(j)) \quad \forall j \in V_c, \forall l \in L_j \quad (31)$$

$$\sum_{l \in L_{ij}} z_{ij}^l = x_{ij} \quad \forall (i, j) \in A \quad (32)$$

$$z_{ij}^l \geq 0 \quad \forall (i, j) \in A, \forall l \in L_{ij} \quad (33)$$

We denote the model CUTR in which the SCF system (17)–(18) is replaced by system (31)–(33) by **LCUTR**, i.e., objective (1), degree constraints (2)–(3), integrality constraints (6), the load-dependent PQ system (31)–(33), and SOP cuts (21). It is easy to argue (as in Gouveia & Voß, 1995) that the LP bound of LCUTR is at least as good as the one of CUTR and in fact the experimental results showed that for many instances it is better. Note that in contrast to the original time-dependent PQ system the load-dependent PQ system alone is not sufficient to eliminate subtours since values ρ_j may also be negative. However, in the model CUTR as well as in the model LCUTR subtour elimination is guaranteed by the SOP cuts (21) (which dominate connection cuts (4)).

5.2. Strengthening the Load-Dependent PQ Model

We can view the load-dependent PQ system (31)–(33) as modeling a path in a layered graph $G_L = (V_L, A_L)$. This layered graph is more complicated than the layered graph corresponding to the original PQ formulation. In G_L a node j_l describes the state when the vehicle leaves node $j \in V$ with load l . Node set $V_L = \{0, n+1\} \cup \{j_l : j \in V, l \in L_j\}$ consists of the start and the end depot, and replicated nodes for all clients for all possible loads. Arc set A_L includes

- start depot arcs $\{(0, j_l) : (0, j) \in A, l = l_{0j} + \rho_j = u_{0j} + \rho_j\}$,
- general arcs $\{(i_l, j_{l+\rho_j}) : (i, j) \in A, i \neq 0, j \neq n+1, l \in L_{ij}\}$, and
- end depot arcs $\{(j_l, n+1) : (j, n+1) \in A, l = l_{j, n+1} = u_{j, n+1}\}$.

This layered graph is reduced by eliminating all nodes except the depot nodes which have no incoming or outgoing arcs since they cannot be part of a feasible solution. An example is shown in Fig.1.

Similar to what has been done in Gouveia et al. (2011) and Godinho et al. (2014) to redefine cut inequalities in the layered graph, we can also redefine the SOP cuts (21) in the load-based layered graph G_L to improve the LP relaxation

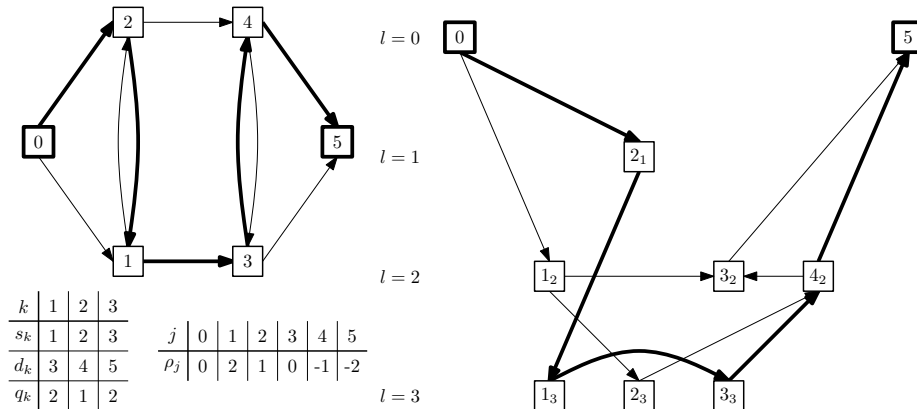


Figure 1: In the left a problem instance with graph G with $n = 4$, a set of commodities, and the aggregated node demands is shown. In the right the corresponding (preprocessed) load-dependent layered graph G_L for vehicle capacity $Q = 3$ is shown. The set of bold arcs in both graphs represent the same feasible solution.

of model LCUTR. Let $S_L := \{j_l \in V_L : j \in S\}$ denote the set of all copies of nodes in some set $S \subseteq V$. The corresponding SOP cuts in G_L are defined as:

$$z(S, V_L^{ij} \setminus S) \geq 1 \quad \forall S \subset V_L^{ij}, \{i\}_L \subseteq S, \{j\}_L \subseteq V_L^{ij} \setminus S, \forall (i, j) \in \tilde{R} \quad (34)$$

These inequalities can be interpreted as the SOP cuts (21) lifted in the load layered graph G_L . It is easy to see that SOP cuts (21) are implied by (34) since the subset of (34) in which all copies of nodes $v \in V^{ij} \setminus \{i, j\}$ belong either to S or to $V_L^{ij} \setminus S$ gives exactly the SOP cuts (21) for a precedence relation $(i, j) \in \tilde{R}$. We denote by **LCUTR₊**, the model LCUTR where (21) are replaced by (34), i.e., objective (1), degree constraints (2)–(3), integrality constraints (6), the load-dependent PQ system (31)–(33), and SOP cuts (34) in G_L . The previous observation implies that the LP relaxation of **LCUTR₊** is not worse than the LP relaxation of LCUTR, and similar to what happens with model CUTR, and as can be seen in Table 2, there is no LP relation between MCF and **LCUTR₊**.

5.3. The Position-Load-Dependent PQ Model

As noted before, the layered graph associated to the load-dependent PQ system is not acyclic. However, the strong models in Godinho et al. (2011, 2014) explicitly use the fact that the associated layered graphs are acyclic (as in the original PQ system) since the layers correspond to the positions of the nodes in the solution. We can derive some information about the position of nodes and arcs based on the given precedence relations. Let $\lambda_{ij}, i \in V \setminus \{n+1\}, j \in V \setminus \{0\}$ be the length of the longest path in precedence graph P from node i to j with respect to the number of arcs. Note that the longest path in an acyclic graph can be computed in time linear in the number of arcs based on topological sorting.

Value $\alpha_j = \lambda_{0j}$ represents a lower bound on the position of node $j \in V_c$ in any feasible tour. Similarly, $\omega_j = n+1 - \lambda_{j,n+1}$ denotes an upper bound on the position of $j \in V_c$ in any feasible tour. Since the positions of the depot nodes are fixed we set $\alpha_0 = \omega_0 = 0$ and $\alpha_{n+1} = \omega_{n+1} = n+1$. Let $P_j := \{\alpha_j, \dots, \omega_j\}$ be the set of possible positions for node $j \in V$. We also define the set of possible positions $P_{ij} := \{\max\{\alpha_i + 1, \alpha_j\}, \dots, \min\{\omega_i + 1, \omega_j\}\}$ for each arc $(i, j) \in A$.

Using this information we propose a combined generalized model that disaggregates variables z_{ij}^l by position. The new variables are defined as z_{ij}^{pl} representing a vehicle on arc (i, j) in position p with load l . The position-load-dependent PQ system is defined as:

$$z^{p, l - \rho_j}(\delta^-(j)) = z^{p+1, l}(\delta^+(j)) \quad \forall j \in V_c, \forall p \in P_j, \forall l \in L_j \quad (35)$$

$$\sum_{p \in P_{ij}} \sum_{l \in L_{ij}} z_{ij}^{pl} = x_{ij} \quad \forall (i, j) \in A \quad (36)$$

$$z_{ij}^{pl} \geq 0 \quad \forall (i, j) \in A, \forall p \in P_{ij}, \forall l \in L_{ij} \quad (37)$$

We observe that we can view these equations in a 3-dimensional layered graph $G_{\text{PL}} = (V_{\text{PL}}, A_{\text{PL}})$ with two resource dimensions, i.e., the position and the load of the vehicle. In G_{PL} we have nodes j_{pl} defining the state when the vehicle arrives at client j on an arc in position p and leaves it with load l . Node set $V_{\text{PL}} = \{0, n+1\} \cup \{j_{pl} : j \in V_c, p \in P_j, l \in L_j\}$ consists of the start and the end depot, and replicated nodes for all clients for all possible positions and loads. Arc set A_{PL} includes

- start depot arcs $\{(0, j_{1l}) : (0, j) \in A, l = l_{0j} + \rho_j = u_{0j} + \rho_j\}$,
- general arcs $\{(i_{p-1, l}, j_{p, l + \rho_j}) : (i, j) \in A, i \neq 0, j \neq n+1, p \in P_{ij}, l \in L_{ij}\}$, and
- end depot arcs $\{(j_{nl}, n+1) : (j, n+1) \in A, l = l_{j, n+1} = u_{j, n+1}\}$.

Note that due to the position dimension, the layered graph G_{PL} is acyclic and thus the generalized PQ system (35)–(37) is sufficient to eliminate subtours.

Similar to what has been suggested in the last subsection we can readapt the SOP cuts (21) in layered graph G_{PL} . Let $S_{\text{PL}} := \{i_{pl} \in V_{\text{PL}} : i \in S\}$ denote the set of all copies of nodes in some set $S \subseteq V$. The corresponding SOP cuts in G_{PL} are defined as:

$$z(S, V_{\text{PL}}^{ij} \setminus S) \geq 1 \quad \forall S \subset V_{\text{PL}}^{ij}, \{i\}_{\text{PL}} \subseteq S, \{j\}_{\text{PL}} \subseteq V_{\text{PL}}^{ij} \setminus S, \forall (i, j) \in \tilde{R} \quad (38)$$

These inequalities can be interpreted as the SOP cuts (21) lifted by exploiting position and load information at the same time. We can use an argument similar to the one in the previous subsection to show that SOP cuts (38) dominate the SOP cuts (34) in the load layered graph G_{L} . We denote the generic model extended by (35)–(37), and (38) by **PLCUTR**₊, i.e., objective (1), degree constraints (2)–(3), integrality constraints (6), the position-load-dependent PQ system (35)–(37), and SOP cuts (38) in G_{PL} . It is easy to argue that the LP relaxation of **PLCUTR**₊ is at least as good as the one of **LCUTR**₊ and our experimental tests indicate that it is significantly better provided that it can be solved within the time limit. However, the relation between the LP bounds of the models **MCF** and **PLCUTR**₊ is still open.

Figure 2 summarizes the strength relations of the LP relaxations of all the models proposed in the last two sections. We note that there is no LP relation between the bounds given by the models **MCF** and **CUTR**, **LCUTR** and **LCUTR**₊. This can be observed from the experimental tests, e.g., in Table 2.

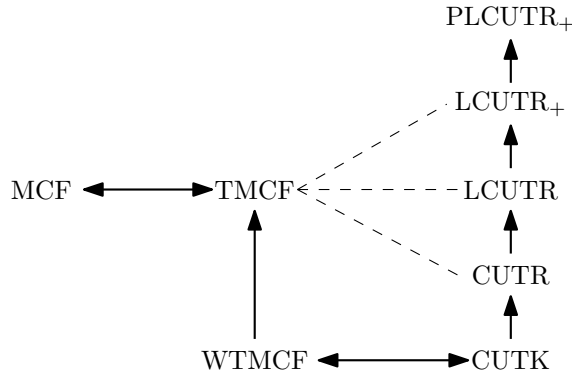


Figure 2: Strength relations: The model at the head of an arrow provides an optimal LP relaxation value which is at least as good as the one of the model at the corresponding tail. Dashed lines indicate that there is no LP relation between the two models.

5.4. Valid Inequalities

Finally, we also consider valid inequalities similar to the ones in Section 4.6 in the variable space defined by the layered graphs G_L and G_{PL} . Note that the concept of predecessors and successors is more complicated in layered graphs since in contrast to original graph G the solution path in G_L or G_{PL} is not Hamiltonian. We know, however, that exactly one of the copies of each original node has to be visited. Thus, in the context of layered graphs a precedence relation $(i, j) \in R$ means that one of the copies of node i has to be visited before one of the copies of node j . However, we cannot say that one particular copy of node i has to be before one particular copy of node j since one or both copies may not be visited at all. Thus, the predecessors and successors of some subset $S \subseteq V_L$ in G_L need to be defined as $\pi_L(S) := \{i_l \in V_L : (i, j) \in R \wedge \{j\}_L \subseteq S\}$ and $\sigma_L(S) := \{j_l \in V_L : (i, j) \in R \wedge \{i\}_L \subseteq S\}$, respectively. The definitions in G_{PL} are similar.

As already shown above SOP inequalities (34) and (38) are lifted variants of SOP inequalities (21) in the layered graph. In a similar way we lift the π -, σ -, and (π, σ) -inequalities by Balas et al. (1995) to the space of variables z_{ij}^l and z_{ij}^p , respectively. When a violated inequality (34) and (38) is found we try to lift it to the transformed π -, σ -, and (π, σ) -inequality in G_L and G_{PL} , respectively, similar to the liftings of (21) mentioned in Section 4.6: We may exclude further cut arcs based on the found cut set while considering the different meaning of predecessors and successors, as mentioned above.

Finally, we also lift inequalities (24) to the space of variables z_{ij}^l . Let $(i_1, i_2), \dots, (i_{k-1}, i_k) \in R, i_1, \dots, i_k \in V \setminus \{n+1\}$, for odd values of $k \geq 3$ be a non-dominated sequence of precedence relations as defined in Subsection 4.6. All copies of odd nodes in G_L are fixed to some set $S \subset V_L \setminus \{n+1\}$ and all copies of even nodes to set $V_L \setminus S$, i.e., $\bigcup_{h \leq k, h \text{ odd}} \{i_h\}_L \subseteq S$ and $\bigcup_{h \leq k, h \text{ even}} \{i_h\}_L \subset V_L \setminus S$. The set of excluded nodes is defined as $S' = \bigcup_{(j, i_1) \in R \vee (i_k, j) \in R \setminus \tilde{R}} \{j\}_L$. Then, we obtain inequalities

$$z(S \setminus S', V_L \setminus (S \cup S')) \geq \lceil k/2 \rceil. \quad (39)$$

Similar to inequalities (24) this lifted variant can be separated in polynomial time by computing the max-flow in a support graph $G_L'' = (V_L' \setminus S', A_L' \setminus \{(i, j) :$

$i \in S' \vee j \in S'\}$) with $V'_L = V_L \cup \{s, t\}$, and $A' = A_L \cup \{(s, i_h) : h \leq k, h \text{ odd}\} \cup \{(i_h, t) : h \leq k, h \text{ even}\}$. The capacities on the arcs incident to node s and t are set to 1. Then, the max-flow from s to t is equivalent to a minimum cut in G_L satisfying the requirements above.

6. Branch-and-Cut Algorithm

The proposed models are solved with a branch-and-cut algorithm based on the framework IBM ILOG CPLEX 12.6. In this section we mention non-default settings of CPLEX, details about the cutting plane algorithm, and methods to obtain primal bounds. Both the cutting plane algorithm and the primal heuristics are provided to CPLEX via callback functions. All settings have been identified in preliminary tests with a diverse subset of the instances. We denote by x^{LP} the solution of the LP relaxation in some branch-and-bound node.

6.1. General Settings

We use default settings for CPLEX with the following exceptions: The solution emphasis is set to “optimality” and general-purpose heuristics are switched off since primal bounds are provided by our own problem-specific heuristics. All variables are declared to be integral since this turned out to be beneficial for the presolving and branching phase of CPLEX, but branching on the x -variables is prioritized.

6.2. Cutting Plane Algorithm

In each cutting plane iteration within a branch-and-bound node we search for violated inequalities of all sets considered in a particular setting. However, to appropriately deal with a possibly large number of added inequalities and slow cutting plane convergence (cf. Uchoa, 2011), we apply the following rules:

- Suppose that a valid inequality in graph G has the form $x(A') \geq b$, then we only add a violated cut if $x^{\text{LP}}(A') < \Delta_G \cdot b$ with $\Delta_G \in (0, 1]$. Similarly, we use parameters Δ_{G_L} and $\Delta_{G_{PL}}$ for valid inequalities in G_L and G_{PL} , respectively.
- If the LP relaxation value does not increase in the last five cutting plane iterations within a branch-and-bound node we continue with branching.
- We add at most 100 violated inequalities per considered set of inequalities within one cutting plane iteration.
- After solving a maximum flow to search for violated cut sets we might obtain multiple minimum cuts. In this case we only consider the minimum cut with the smallest and the largest set S , and only add the cut inequality for which the number of cut arcs is minimal.

Together with the exact separation algorithms described in the previous sections, we apply in each cutting plane iteration the heuristic by Hernández-Pérez & Salazar-González (2009) to identify further violated inequalities: Essentially, we perform a restricted enumeration of node sets S and check for violated π -, σ -, and (π, σ) -inequalities, and capacity cuts (30).

Algorithm 1: Primal Heuristic

Input: graph G , LP solution x^{LP} , global best solution S_g
Output: solution S for the m -PDTSP (empty if none can be found)
// Solution construction

```
1  $N = 1, S = \emptyset$ 
2 while ( $S$  is infeasible or duplicate) and  $N \leq 10$  do
3    $S = \{0\}$  // sequence of nodes starting in depot 0
4   while  $|S| < |V|$  and  $S$  can be feasibly extended do
5     extend  $S$  by a node chosen randomly from the  $N$  cheapest feasible
     successors (based on costs  $c'_{ij} = c_{ij}(1 - x_{ij}^{\text{LP}}), \forall (i, j) \in A$ )
6    $N = N + 1$ 
7 if  $S$  is infeasible then return  $\emptyset$ 
   // Solution improvement
8  $I = 0, N_s = 2$ 
9 while  $I < 30$  do
10   $S' = S_g$  (with  $\mathcal{P} = 50\%$ ), else  $S$ 
11  apply to  $S'$  consecutively  $N_s$  random feasible node shifts
12  while  $S'$  can be improved do
13    apply to  $S'$  randomly one of the ten most improving moves out of
    all feasible node shifts and node swaps
14  if  $c(S') < c(S)$  then  $S = S', I = 0, N_s = 2$ 
15  else  $I = I + 1, N_s = \min\{N_s + 1, 10\}$ 
16 return  $S$ 
```

6.3. Primal Heuristics

Since primal bounds are essential for pruning the branch-and-bound tree and fixing variables based on reduced costs we also use heuristics in each of the branch-and-bound nodes. These heuristics are called after each cut iteration in the root node of the branch-and-bound tree, in every 5th branch-and-bound node within the first 100 nodes, in every 25th node within the first 1000 nodes, and in every 50th node in the rest of the nodes. In the remaining of this subsection we give a brief overview of the heuristics that we use (see also Algorithm 1).

To construct a feasible solution we apply a nearest neighbor heuristic (Rodríguez-Martín & Salazar-González, 2012) guided by the LP solution of the current branch-and-bound node in the sense that we use modified arc costs $c'_{ij} = c_{ij}(1 - x_{ij}^{\text{LP}})$ for each $(i, j) \in A$: The solution path is extended step-by-step by choosing the cheapest unvisited successor node without violating the vehicle capacity and the precedence relations. We store all solutions in a hash-based archive to prevent duplicates. If we are not able to construct a feasible solution or if we obtain a duplicate we start again in a GRASP manner (Feo & Resende, 1995), i.e., we randomly choose among the N cheapest extension nodes, with N being increased from 2 to 10 in case of infeasibility. If after ten tries we obtain no feasible solution we continue with the branch-and-cut algorithm.

To further improve a created solution, we run a generalized variable neighborhood search (GVNS) (Hansen & Mladenović, 2001). We stop the GVNS if after 30 iterations no new global best solution can be found. With a probability

of 50% we choose the global best solution for a GVNS iteration, otherwise we use the best solution in the current heuristic call. To locally improve the solution an embedded variable neighborhood descent (VND) based on two neighborhood structures is applied: i) One node is shifted to another position in the path, and ii) two nodes are swapped. In each iteration in the VND we choose randomly among the ten most improving feasible moves from both neighborhoods. To diversify the solution in the shaking phase of the GVNS we apply two random node shifts. If after a GVNS iteration no new global best solution can be found the number of shaking moves is increased by one (up to at most 10).

7. Experiments

This section reports and discusses experimental results for instances of the m -PDTSP and the SOP. Each test run was performed on a single core of an Intel Xeon E5540 or E5649 machine both with 2.53 GHz. Preliminary tests showed that both machines have nearly the same performance with respect to our type of experiments. The memory limit per test run was set to 8 GB. The CPU times for the preprocessing from Section 2 is included in all given running times.

7.1. Results for the m -PDTSP

The maximum CPU time to obtain the optimal solutions for the integer models and the respective LP relaxations of the m -PDTSP was set to 7200 seconds. We used three different classes of instances introduced by Hernández-Pérez & Salazar-González (2009): Class 1 has been derived from instances for the SOP, each precedence relation corresponding to a commodity with demand 1 (Suffix “max1”) or with a randomly chosen demand in $\{1, \dots, 5\}$ (Suffix “max5”). Class 2 and 3 have n points randomly placed in a square with costs corresponding to the Euclidian distances and different numbers of commodities with randomly chosen origin, destination, and demand in $\{1, \dots, 5\}$. The difference between the last two classes is that in class 3 each node is the origin or destination of exactly one commodity whereas in class 2 this restriction does not hold. Class 1 are single instances whereas the other two classes contain sets of ten instances with the same general properties (number of nodes, number of commodities, vehicle capacity). We only considered instances from these sets which are not shown to be unconstrained or infeasible in the preprocessing phase with respect to the associated vehicle capacity.

Tables 2 and 3 compare the LP relaxations of the different models shown in Fig. 2. Additionally, we enhance model CUTR by considering all valid inequalities described in Section 4.6, and denote it by CUTR*. Similarly, we denote model LCUTR₊ with all valid inequalities in Section 4.6 and 5.4 by LCUTR*₊, and PLCUTR₊ with the same inequalities formulated in graph G_{PL} instead of G_L by PLCUTR*₊. For the last three models (with suffix *) we also perform heuristic separation as described in Section 6.2. Because of this and the heuristic liftings the LP relations to the models with exact deterministic separation are not consistent. Similarly, the Benders decomposition approach (BE) based on the MCF model by Hernández-Pérez & Salazar-González (2009) (HS) also contains heuristic elements. We adopted all results of BE from Hernández-Pérez & Salazar-González (2009). In the cutting plane algorithm for computing the LP

Table 2: Comparison of LP relaxations of different models for class 1 instances. Bold values denote the best LP gaps.

Instance	V	K	Q	LP gap in %										LP time in seconds																										
				HS		CUT-		LCUT-		PLCUT-		HS		CUT-		LCUT-		PLCUT-																						
				BE	MCF	K	R	R*	R	R ₊	R ₊ *	R ₊	R ₊ *	BE	MCF	K	R	R*	R	R ₊	R ₊ *	R ₊	R ₊ *																	
ESC07Q3max1	9	6	3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
ESC07Q10max5	9	6	10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
ESC12Q4max1	14	7	4	17.8	17.8	18.4	18.4	18.4	14.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
ESC12Q5max1	14	7	5	13.8	13.9	14.0	13.9	13.9	12.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
ESC12Q15max5	14	7	15	13.8	14.0	14.0	13.9	13.9	12.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
ESC25Q3max1	27	9	3	12.2	10.1	12.5	11.6	9.4	11.2	7.0	6.9	5.1	5.1	0	1	0	0	0	0	0	0	1	0	0	0	0	1	1	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl		
ESC25Q4max1	27	9	4	9.0	14.7	15.6	13.5	9.6	10.3	7.4	6.9	5.0	5.0	0	1	0	0	0	0	0	0	1	0	0	0	0	2	2	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl		
ESC25Q5max1	27	9	5	0.6	4.1	4.1	2.3	0.0	2.3	0.0	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	146	139	tl	tl	tl	tl	tl	tl	tl	tl		
ESC25Q15max5	27	9	15	1.2	4.1	4.1	2.3	1.2	2.1	0.0	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	3	931	922	tl	tl	tl	tl	tl	tl	tl	tl		
ESC25Q20max5	27	9	20	2.1	4.1	4.1	2.3	1.0	2.1	0.0	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	5	1582	1585	tl	tl	tl	tl	tl	tl	tl	tl		
ESC47Q3max1	49	10	3	7.6	6.6	7.0	7.0	6.4	7.0	5.0	4.9	6.9	6.8	0	10	2	1	1	2	7	11	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl		
ESC47Q4max1	49	10	4	4.0	3.2	3.2	3.2	2.6	3.2	1.8	1.7	5.0	3.0	0	8	2	1	2	4	15	27	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl		
ESC47Q10max5	49	10	10	7.6	7.0	7.0	7.0	6.4	7.0	4.3	4.3	100.0	100.0	0	8	2	1	2	9	179	173	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
ESC47Q15max5	49	10	15	4.0	3.2	3.2	3.2	2.6	3.2	1.6	1.6	100.0	100.0	0	8	1	1	1	14	261	238	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
ESC47Q20max5	49	10	20	4.0	3.2	3.2	3.2	2.6	3.2	1.6	1.6	100.0	100.0	0	8	1	1	2	17	346	260	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl
br17.10Q3max1	18	10	3	24.0	20.7	32.9	31.7	23.2	31.7	25.6	18.3	2.4	2.4	0	1	0	0	0	0	0	0	0	0	0	0	2	2	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
br17.10Q4max1	18	10	4	24.7	32.9	41.1	39.7	24.7	37.0	30.1	21.9	17.8	17.8	0	1	0	0	0	0	0	0	0	0	0	0	2	2	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
br17.10Q5max1	18	10	5	0.0	20.0	25.5	21.8	0.0	21.8	14.5	0.0	3.6	0.0	0	1	0	0	0	0	0	0	0	0	0	8	0	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
br17.10Q10max5	18	10	10	16.1	22.7	34.8	30.3	16.7	28.8	12.1	7.6	7.6	6.1	0	1	0	0	0	0	0	0	0	0	0	28	75	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
br17.10Q15max5	18	10	15	0.0	23.6	25.5	21.8	0.0	21.8	7.3	0.0	5.5	0.0	0	0	0	0	0	0	0	0	0	0	0	1082	1	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl
br17.12Q3max1	18	12	3	47.6	42.9	53.8	52.9	44.5	52.1	42.9	37.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3196	3952	tl	tl	tl	tl	tl	tl	tl	tl		
br17.12Q4max1	18	12	4	25.7	33.8	43.2	40.5	25.7	36.5	24.3	18.9	13.5	13.5	0	1	0	0	0	0	0	0	0	0	0	2	3	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
br17.12Q5max1	18	12	5	0.0	20.0	25.5	21.8	0.0	21.8	12.7	0.0	0.0	0.0	0	1	0	0	0	0	0	0	0	0	0	1	0	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
br17.12Q10max5	18	12	10	25.7	33.8	40.5	36.5	25.7	35.1	13.5	12.2	1.4	1.4	0	1	0	0	0	0	0	0	0	0	0	50	60	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
br17.12Q15max5	18	12	15	9.1	25.5	25.5	21.8	9.1	21.8	5.5	1.8	3.6	3.6	0	0	0	0	0	0	0	0	0	0	0	1	1264	4741	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl
p43.1Q2max1	44	9	2	-	-	-	-	-	-	-	-	-	-	-	37	2	2	2	4	220	751	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
p43.1Q3max1	44	9	3	-	-	-	-	-	-	-	-	-	-	-	40	2	2	2	3	1754	1218	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl
p43.1Q4max1	44	9	4	-	-	-	-	-	-	-	-	-	-	-	60	2	1	3	8	1544	2286	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl
p43.1Q10max5	44	9	10	-	-	-	-	-	-	-	-	-	-	-	62	1	2	2	12	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
p43.1Q15max5	44	9	15	-	-	-	-	-	-	-	-	-	-	-	29	1	2	3	55	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
p43.2Q10max1	44	20	10	-	-	-	-	-	-	-	-	-	-	-	1154	3	3	4	64	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
p43.2Q40max5	44	20	40	-	-	-	-	-	-	-	-	-	-	-	223	2	2	4	964	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
p43.3Q10max1	44	37	10	-	-	-	-	-	-	-	-	-	-	-	2837	4	7	8	63	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
p43.3Q40max5	44	37	40	-	-	-	-	-	-	-	-	-	-	-	1006	3	5	8	1750	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
p43.4Q10max1	44	50	10	-	-	-	-	-	-	-	-	-	-	-	34	1	1	3	15	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	
p43.4Q40max5	44	50	40	-	21.5	21.5	16.2	0.1	16.2	16.2	0.1	100.0	100.0	-	17	1	1	2	265	tl	4296	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	tl	

relaxation we set $\Delta_G = \Delta_{G_L} = \Delta_{G_{PL}} = 1$ and do not perform early branching to obtain the correct LP relaxation value. Let c_{OPT} be the optimal integer solution value and c_{LP} be the optimal value of the LP relaxation. The LP gap value for one particular model and instance in the tables is given by $(c_{OPT} - c_{LP})/c_{OPT}$. If value c_{OPT} or c_{LP} is not available we skip the corresponding LP gap value (“-” in the tables). The CPU times do not involve instances which are determined to be infeasible after solving the LP relaxation. If all instances of a set are infeasible we write “inf” in the tables. If the time limit is reached before the cutting plane algorithm was finished we write “tl” in the tables or use 7200 seconds to compute the average values.

By aggregating the commodities in model CUTK we lose some information about the demand structure which can be observed in the weaker gaps with respect to model MCF. However, by adding further valid inequalities in CUTR and CUTR* this disadvantage can be alleviated for most of the instances, except for those with very tight vehicle capacities. It can be clearly seen that the layered graph models obtain significantly smaller LP gaps than the other models defined on the original graph. However, for several instances it was not possible to compute the optimal LP relaxation value within the time limit, especially for large models on the 3-dimensional layered graph. Note that we observed that for some instances infeasibility can be shown in the LP relaxation only with the strong models.

Tables 4 and 5 show the results of our branch-and-cut algorithms in comparison to the Benders decomposition approach (BE) by Hernández-Pérez & Salazar-González (2009). Here, we only consider a subset of our models, namely CUTR* (C), LCUTR*₊ (L), and PLCUTR*₊ (PL). In the embedded cutting plane algorithms we set $\Delta_G = 0.75$, $\Delta_{G_L} = \Delta_{G_{PL}} = 0.25$. Let c_{LB} and c_{UB} be

Table 4: Comparison of branch-and-cut algorithms based on different models for class 1 instances (C ... CUTR*, L ... LCUTR*, PL ... PLCUTR*). Bold values denote the best CPU times.

Instance	V	K	Q	removed arcs in %	gap in %			time in sec.				infeasible			time limit			
					C	L	PL	BE	C	L	PL	BE	C	L	PL	BE	C	L
ESC07Q3max1	9	6	3	18	0.0	0.0	3.5	0	0	0	0	0	0	0	0	0	0	0
ESC07Q10max5	9	6	10	18	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0
ESC12Q4max1	14	7	4	11	0.0	0.0	0.0	0	0	0	3	0	0	0	0	0	0	0
ESC12Q5max1	14	7	5	10	0.0	0.0	0.0	0	1	0	1	0	0	0	0	0	0	0
ESC12Q15max5	14	7	15	10	0.0	0.0	0.0	0	1	0	2	0	0	0	0	0	0	0
ESC25Q3max1	27	9	3	4	0.0	0.0	0.0	43	13	10	1291	0	0	0	0	0	0	0
ESC25Q4max1	27	9	4	3	0.0	0.0	0.0	5	7	10	1063	0	0	0	0	0	0	0
ESC25Q5max1	27	9	5	3	0.0	0.0	0.0	0	1	2	104	0	0	0	0	0	0	0
ESC25Q15max5	27	9	15	3	0.0	0.0	0.0	0	1	2	886	0	0	0	0	0	0	0
ESC25Q20max5	27	9	20	3	0.0	0.0	0.0	0	1	6	1497	0	0	0	0	0	0	0
ESC47Q3max1	49	10	3	2	0.0	0.0	47.1	61	40	175	tl	0	0	0	0	0	0	1
ESC47Q4max1	49	10	4	2	0.0	0.0	56.4	12	12	22	tl	0	0	0	0	0	0	1
ESC47Q10max5	49	10	10	2	0.0	0.0	-	61	83	1706	tl	0	0	0	0	0	0	1
ESC47Q15max5	49	10	15	2	0.0	0.0	-	10	12	403	tl	0	0	0	0	0	0	1
ESC47Q20max5	49	10	20	2	0.0	0.0	-	10	17	311	tl	0	0	0	0	0	0	1
br17.10Q3max1	18	10	3	16	0.0	0.0	0.0	28	7	4	126	0	0	0	0	0	0	0
br17.10Q4max1	18	10	4	11	0.0	0.0	0.0	6868	2284	279	6355	0	0	0	0	0	0	0
br17.10Q5max1	18	10	5	8	0.0	0.0	0.0	0	0	0	13	0	0	0	0	0	0	0
br17.10Q10max5	18	10	10	14	0.0	0.0	0.0	73	25	23	802	0	0	0	0	0	0	0
br17.10Q15max5	18	10	15	8	0.0	0.0	0.0	0	0	1	40	0	0	0	0	0	0	0
br17.12Q3max1	18	12	3	23	0.0	0.0	0.0	1820	57	18	515	0	0	0	0	0	0	0
br17.12Q4max1	18	12	4	14	0.0	0.0	0.0	3049	888	27	5265	0	0	0	0	0	0	0
br17.12Q5max1	18	12	5	11	0.0	0.0	0.0	0	0	0	5	0	0	0	0	0	0	0
br17.12Q10max5	18	12	10	19	0.0	0.0	0.0	2040	191	58	1735	0	0	0	0	0	0	0
br17.12Q15max5	18	12	15	11	0.0	0.0	0.0	1	1	5	186	0	0	0	0	0	0	0
p43.1Q2max1	44	9	2	3	48.7	48.7	49.0	-	tl	tl	tl	-	0	0	0	-	1	1
p43.1Q3max1	44	9	3	1	0.4	0.4	1.5	-	tl	tl	tl	-	0	0	0	-	1	1
p43.1Q4max1	44	9	4	1	0.0	0.1	48.9	-	tl	tl	tl	-	0	0	0	-	1	1
p43.1Q10max5	44	9	10	1	0.0	0.3	-	-	tl	tl	tl	-	0	0	0	-	1	1
p43.1Q15max5	44	9	15	1	0.1	0.2	-	-	tl	tl	tl	-	0	0	0	-	1	1
p43.2Q10max1	44	20	10	3	0.4	0.6	-	-	tl	tl	tl	-	0	0	0	-	1	1
p43.2Q40max5	44	20	40	3	0.4	0.6	-	-	tl	tl	tl	-	0	0	0	-	1	1
p43.3Q10max1	44	37	10	6	1.2	1.7	-	-	tl	tl	tl	-	0	0	0	-	1	1
p43.3Q40max5	44	37	40	6	0.4	0.7	-	-	tl	tl	tl	-	0	0	0	-	1	1
p43.4Q10max1	44	50	10	39	0.1	0.3	-	-	tl	tl	tl	-	0	0	0	-	1	1
p43.4Q40max5	44	50	40	39	0.0	0.0	-	-	12	tl	tl	-	0	0	0	-	0	1

the best global lower and upper bounds, respectively, obtained by the algorithm within the time limit. The gaps in the tables are given by $(c_{UB} - c_{LB})/c_{UB}$. If at least one of the bounds is not available we skip the corresponding gap value (“-” in the tables). Note that these gaps are not available for the BE approach. Again, the CPU times do not involve instances which are shown to be infeasible in the solution process. If the time limit is reached before proving optimality we write “tl” in the tables or use 7200 seconds to compute the average values. Additionally, the tables include the number of instances which are shown to be infeasible and the number of instances for which the algorithm reaches the time limit. Note that the CPU times of BE have been obtained on a different hardware with CPLEX 10.2. The column “removed arcs in %” shows the average percentage of arcs which are eliminated in the preprocessing phase described in Section 2.

For class 2 and 3 instances with a large number of commodities the demand aggregation discussed in Section 4 (model CUTR*) is quite beneficial in terms of lower CPU times when compared to the BE approach. Additionally, we are able to solve several open m -PDTSP instances. The branch-and-cut algorithm based on LCUTR*₊ shows significant improvements on instances with extremely tight vehicle capacities (cf. br17.10, br17.12 in Table 4). However, both layered graph variants are not competitive on larger instances because of the large size of the corresponding models.

7.2. Results for the SOP

As mentioned before, relaxing the capacity constraints in the m -PDTSP leads to the SOP. Therefore, we also provide branch-and-cut results on benchmark instances for the SOP. The CPU time limit is extended to 1 day. We re-

Table 5: Comparison of branch-and-cut algorithms based on different models for class 2 and 3 instances (C ... CUTR*, L ... LCUTR*, PL ... PLCUTR*). Bold values denote the best CPU times. Each set contains 10 instances.

Set	V	K	Q	removed arcs in %			avg. gap in %			avg. time in sec.				# infeasible			# time limit		
				C	L	PL	BE	C	L	PL	BE	C	L	PL	BE	C	L	PL	
n10m5Q10	11	5	10	23	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	
n10m5Q15	11	5	15	21	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	
n10m5Q20	11	5	20	21	0.0	0.0	0.0	-	0	0	0	-	0	0	0	-	0	0	
n10m10Q10	11	10	10	89	0.0	0.0	0.0	0	0	0	0	6	7	7	7	0	0	0	
n10m10Q15	11	10	15	57	0.0	0.0	0.0	0	0	0	0	1	1	1	1	0	0	0	
n10m10Q20	11	10	20	51	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	
n10m10Q25	11	10	25	51	0.0	0.0	0.0	-	0	0	0	-	0	0	0	-	0	0	
n10m10Q30	11	10	30	51	0.0	0.0	0.0	-	0	0	0	-	0	0	0	-	0	0	
n10m15Q10	11	15	10	100	-	-	-	-	-	-	-	-	10	10	10	-	0	0	
n10m15Q15	11	15	15	99	0.0	0.0	0.0	0	0	0	0	9	9	9	9	0	0	0	
n10m15Q20	11	15	20	92	0.0	0.0	0.0	0	0	0	0	6	6	6	6	0	0	0	
n10m15Q25	11	15	25	85	0.0	0.0	0.0	0	0	0	0	4	4	4	4	0	0	0	
n10m15Q30	11	15	30	77	0.0	0.0	0.0	0	0	0	0	2	2	2	2	0	0	0	
n15m5Q10	16	5	10	9	0.0	0.0	0.0	0	1	4	70	0	0	0	0	0	0	0	
n15m5Q15	16	5	15	8	0.0	0.0	0.0	0	1	16	144	0	0	0	0	0	0	0	
n15m5Q20	16	5	20	8	0.0	0.0	0.0	-	1	11	246	-	0	0	0	-	0	0	
n15m10Q10	16	10	10	35	0.0	0.0	0.0	1801	1	4	74	6	7	7	7	1	0	0	
n15m10Q15	16	10	15	23	0.0	0.0	0.0	0	1	3	36	1	1	1	1	0	0	0	
n15m10Q20	16	10	20	20	0.0	8.0	0.0	0	0	6	48	0	0	0	0	0	0	0	
n15m10Q25	16	10	25	20	0.0	0.0	0.0	0	0	7	74	0	0	0	0	0	0	0	
n15m10Q30	16	10	30	20	0.0	0.0	0.0	0	0	7	67	-	0	0	0	-	0	0	
n15m15Q10	16	15	10	64	-	-	-	-	-	-	-	-	10	10	10	-	0	0	
n15m15Q15	16	15	15	47	0.0	0.0	0.0	2	1	3	21	4	4	4	4	0	0	0	
n15m15Q20	16	15	20	39	0.0	0.0	0.0	1	0	0	4	2	2	2	2	0	0	0	
n15m15Q25	16	15	25	37	0.0	0.0	0.0	0	0	0	5	0	0	0	0	0	0	0	
n15m15Q30	16	15	30	37	0.0	0.0	0.0	0	0	0	6	0	0	0	0	0	0	0	
n20m5Q10	21	5	10	5	0.0	0.0	0.0	3	1	2	120	0	0	0	0	0	0	0	
n20m5Q15	21	5	15	4	0.0	0.0	0.0	0	0	1	167	0	0	0	0	0	0	0	
n20m5Q20	21	5	20	4	0.0	0.0	0.0	-	0	2	167	-	0	0	0	-	0	0	
n20m10Q10	21	10	10	15	13.0	13.3	14.8	1832	1806	1854	3475	2	2	2	2	2	2	3	
n20m10Q15	21	10	15	10	0.0	0.0	2.2	67	31	374	3138	0	0	0	0	0	0	3	
n20m10Q20	21	10	20	9	0.0	0.0	0.8	53	1	156	1853	0	0	0	0	0	0	2	
n20m10Q25	21	10	25	9	0.0	0.0	0.8	53	1	231	1770	0	0	0	0	0	0	2	
n20m10Q30	21	10	30	9	0.0	0.0	0.9	-	1	212	1835	-	0	0	0	-	0	2	
n20m15Q10	21	15	10	42	-	-	-	-	tl	tl	tl	-	8	8	8	-	2	2	
n20m15Q15	21	15	15	23	22.3	23.4	28.6	5305	3399	3684	6304	0	0	0	0	7	4	8	
n20m15Q20	21	15	20	19	0.6	2.1	5.5	3073	910	2239	4864	0	0	0	0	4	1	6	
n20m15Q25	21	15	25	18	0.0	0.0	3.2	172	6	532	3397	0	0	0	0	0	0	4	
n20m15Q30	21	15	30	18	0.0	0.0	0.9	114	2	278	2555	0	0	0	0	0	0	2	
n25m5Q10	26	5	10	3	0.0	0.0	11.3	2	10	25	2784	0	0	0	0	0	0	3	
n25m5Q15	26	5	15	3	0.0	0.0	8.2	1	2	39	2271	0	0	0	0	0	0	1	
n25m5Q20	26	5	20	3	0.0	0.0	0.7	1	2	44	2382	0	0	0	0	0	0	1	
n25m10Q10	26	10	10	8	0.9	1.3	9.6	3684	2004	3916	6545	1	1	1	1	3	2	8	
n25m10Q15	26	10	15	6	0.0	0.0	6.3	137	67	1577	tl	0	0	0	0	0	0	10	
n25m10Q20	26	10	20	6	0.0	0.4	4.1	14	5	1980	6631	0	0	0	0	0	2	9	
n25m10Q25	26	10	25	6	0.0	0.5	4.6	14	4	2367	6466	0	0	0	0	0	3	8	
n25m10Q30	26	10	30	6	0.0	0.4	4.1	-	4	2146	6697	-	0	0	0	-	0	1	
n25m15Q10	26	15	10	24	61.5	61.9	67.9	-	tl	tl	tl	-	3	3	3	-	7	7	
n25m15Q15	26	15	15	13	4.5	7.3	16.4	5786	3167	5333	tl	0	0	0	0	8	4	10	
n25m15Q20	26	15	20	10	0.7	3.8	9.2	3804	1385	4787	6520	0	0	0	0	5	1	6	
n25m15Q25	26	15	25	10	0.0	1.5	6.1	1387	59	3545	6864	0	0	0	0	1	0	3	
n25m15Q30	26	15	30	10	0.0	1.8	6.5	565	14	3388	tl	0	0	0	0	0	4	10	
m5Q5	12	5	5	43	0.0	0.0	0.0	0	0	0	0	0	0	0	0	0	0	0	
m5Q10	12	5	10	14	0.0	0.0	0.0	0	0	0	1	0	0	0	0	0	0	0	
m5Q15	12	5	15	14	0.0	0.0	0.0	0	0	0	1	0	0	0	0	0	0	0	
m10Q5	22	10	5	48	0.0	0.0	0.0	2	2	1	93	0	0	0	0	0	0	0	
m10Q10	22	10	10	7	0.0	0.0	4.6	87	165	612	5670	0	0	0	0	0	0	7	
m10Q15	22	10	15	7	0.0	0.2	5.2	62	30	1741	5122	0	0	0	0	0	2	7	
m10Q20	22	10	20	7	0.0	0.0	2.6	2	2	328	4128	0	0	0	0	0	0	5	
m10Q25	22	10	25	7	0.0	0.1	2.3	1	2	1099	4086	0	0	0	0	0	1	4	
m10Q30	22	10	30	7	0.0	0.1	2.1	1	2	1294	4379	0	0	0	0	0	1	4	
m15Q5	32	15	5	43	1.1	1.1	12.0	2006	2529	1053	5922	0	0	0	0	2	1	8	
m15Q10	32	15	10	5	6.2	9.1	18.9	6523	6493	6908	tl	0	0	0	0	9	9	10	
m15Q15	32	15	15	5	2.4	7.5	15.1	4124	3284	6595	tl	0	0	0	0	5	4	10	
m15Q20	32	15	20	5	0.0	5.6	43.5	918	269	7033	tl	0	0	0	0	0	0	10	
m15Q25	32	15	25	5	0.0	4.4	92.3	118	40	5971	tl	0	0	0	0	0	8	10	
m15Q30	32	15	30	5	0.0	6.0	-	101	43	6482	tl	0	0	0	0	0	0	10	

Table 6: Comparison of branch-and-cut algorithms for SOP instances from the TSPLIB. Bold instance names mark instances solved for the first time. Bold bounds and CPU times denote the best results.

Instance	V	A	\bar{R}	LB		UB		time in seconds					
				BK	BCx	BK	BCx	BK	BC1	BC2	BC3	BC4	
ESC07	9	40	6	2125	2125	2125	2125	0	0	0	0	0	0
ESC12	14	132	7	1675	1675	1675	1675	0	0	0	0	0	0
ESC25	27	622	9	1681	1681	1681	1681	1	0	0	0	0	0
ESC47	49	2187	10	1288	1288	1288	1288	28	7	8	4	2	2
ESC63	65	3613	95	62	62	62	62	0	9	2	1	1	1
ESC78	80	5550	77	18230	18230	18230	18230	1	770	46	8664	7569	7569
br17.10	18	237	10	55	55	55	55	0	1	0	1	0	0
br17.12	18	223	12	55	55	55	55	0	1	0	1	0	0
ft53.1	54	2722	12	7531	7531	7531	7531	6768	183	218	91	145	145
ft53.2	54	2680	25	7630	8026	8026	8026	-	29842	-	-	-	-
ft53.3	54	2306	48	9473	10262	10262	10262	-	15693	8629	-	-	-
ft53.4	54	1218	63	14425	14425	14425	14425	121	11	2	5	5	5
ft70.1	71	4783	17	39313	39313	39313	39313	363	27	48	17	18	18
ft70.2	71	4714	35	39843	40101	40419	40728	-	-	-	-	-	-
ft70.3	71	4384	68	41413	42535	42535	42535	-	61197	28691	-	-	-
ft70.4	71	2154	86	53072	53530	53530	53530	-	769	315	308	249	249
krol24p.1	101	9814	25	37861	38762	39420	39420	-	-	-	-	-	-
krol24p.2	101	9738	49	38809	39841	41336	41336	-	-	-	-	-	-
krol24p.3	101	9339	97	41578	43904	49499	49570	-	-	-	-	-	-
krol24p.4	101	5260	131	65445	73021	76103	76103	-	-	-	-	-	-
p43.1	44	1778	9	28140	28140	28140	28140	288	4	2	6	5	5
p43.2	44	1724	20	28480	28375	28480	28480	279	-	-	-	-	-
p43.3	44	1600	37	28835	28766	28835	28835	177	-	-	-	-	-
p43.4	44	795	50	83005	83005	83005	83005	88	35	11	11	13	13
prob.42	42	1596	10	243	243	243	243	145	6	15	8	9	9
prob.100	100	9579	41	1027	1045	1163	1346	-	-	-	-	-	-
rbg048a	50	1569	192	351	351	351	351	21	1	1	1	1	0
rbg050c	52	1703	256	467	467	467	467	3	1	1	1	1	1
rbg109a	111	1748	622	1038	1038	1038	1038	13979	2	1	2	2	2
rbg150a	152	2647	952	1748	1750	1750	1750	-	4	2	2	2	2
rbg174a	176	3309	1113	2033	2033	2033	2033	632	9	11	6	6	6
rbg253a	255	5125	1721	2940	2950	2950	2950	-	122	107	22	16	16
rbg323a	325	10021	2412	3137	3140	3140	3140	-	1714	745	159	193	193
rbg341a	343	9884	2542	2543	2568	2568	2568	-	-	-	70997	-	-
rbg358a	360	17998	3239	2529	2545	2545	2545	-	20127	12504	2179	791	791
rbg378a	380	18412	3069	2771	2809	2816	2816	-	-	-	-	-	-
ry48p.1	49	2222	11	15805	15805	15805	15805	12483	-	-	-	-	27300
ry48p.2	49	2188	23	15747	16074	16666	16666	-	-	-	-	-	-
ry48p.3	49	1973	42	18156	19490	19894	19894	-	-	-	-	-	-
ry48p.4	49	1046	58	31446	31446	31446	31446	97	306	92	382	234	234

moved all parts from model CUTR* which are only relevant in the capacitated case, i.e., the flow system with the f -variables. The load-dependent models LCUTR*₊ and PLCUTR*₊ do not make sense for the SOP since they mainly focus on modeling the vehicle capacity. A position-dependent model as shown e.g., in Godinho et al. (2011, 2014) may be appropriate to model the precedence relations. In preliminary tests on large benchmark instances we, however, observed much longer running times compared to the modified CUTR* model. Thus, we only report results for the model CUTR*.

Table 6 shows branch-and-cut results for instances for the SOP from the TSPLIB. The best known (BK) lower and upper bounds (LB,UB) and fastest solution times are obtained from different articles (Ascheuer, 1995; Ascheuer et al., 2000; Gambardella & Dorigo, 2000; Gouveia & Pesneau, 2006; Anghinolfi et al., 2011; Cire & Hoes, 2013). Note that the BK results are obtained on different hardware so they are not directly comparable to our CPU times. Dashes “-” in the tables mean a reached time or memory limit. We compare four different branch-and-cut configurations BC1-4: The heuristic separation and inequalities (24) are only active in BC1-2, we set $\Delta_G = 0.5$ for BC1/3 and $\Delta_G = 0.9$ for BC2/4. Lower and upper bounds BCx are the best over all four branch-and-cut algorithms.

Our branch-and-cut algorithms were able to solve 9 instances for the first time (instance names marked bold in Table 6) and to significantly improve the lower bounds of the residual 9 open instances. Even large instances with hundreds of nodes could be solved to optimality (“rbg”-instances). Inequalities (24) used in BC1-2 are able to close the gap for instances with a large number of

Table 7: Comparison of branch-and-cut algorithms to MO by Montemanni et al. (2013) for SOPLIB instances. Bold instance names mark instances solved for the first time. Bold bounds and CPU times denote the best results. (The UB of MO for instance R.400.1000.15 seems to be wrong.)

Instance	A	$ \bar{R} $	LB			UB			time in seconds		
			MO	BC	BCH	MO	BC	BCH	MO	BC	BCH
R.200.100.1	39402	0	61	61	61	61	61	61	426	28	88
R.200.100.15	7089	991	1257	1560	1585	1792	3111	2003	-	-	-
R.200.100.30	2338	604	4185	4216	4216	4216	4216	4216	-	14	20
R.200.100.60	690	336	71749	71749	71749	71749	71749	71749	0	0	1
R.200.1000.1	39402	0	1404	1404	1404	1404	1404	1404	169	42	628
R.200.1000.15	6315	1005	14565	18741	18936	20481	27598	21393	-	-	-
R.200.1000.30	2286	600	40170	41196	41196	41196	41196	41196	-	9	14
R.200.1000.60	786	327	71556	71556	71556	71556	71556	71556	2	0	0
R.300.100.1	89102	0	26	26	26	26	26	26	2240	199	521
R.300.100.15	10254	1742	2166	2690	2802	3161	-	3355	-	-	-
R.300.100.30	3722	982	5839	6120	6120	6120	6120	6120	-	1366	2411
R.300.100.60	1066	500	9726	9726	9726	9726	9726	9726	1	1	1
R.300.1000.1	89102	0	1294	1294	1294	1294	1294	1294	19864	258	1581
R.300.1000.15	10191	1653	21096	26650	26940	29183	43873	31291	-	-	-
R.300.1000.30	4094	971	51495	54147	54147	54147	54147	54147	-	37	60
R.300.1000.60	1083	498	109471	109471	109471	109471	109471	109471	2	1	1
R.400.100.1	158802	0	13	13	13	13	13	13	4822	113	944
R.400.100.15	14006	2311	2747	3414	3516	3906	-	4228	-	-	-
R.400.100.30	4708	1253	7755	8165	8165	8165	8165	8165	-	12	28
R.400.100.60	1361	662	15228	15228	15228	15228	15228	15228	49	3	3
R.400.1000.1	158802	0	1343	1343	1343	1343	1343	1343	3004	56	720
R.400.1000.15	13564	2389	28159	35103	35364	<i>29685</i>	50600	43268	-	-	-
R.400.1000.30	4868	1238	79868	85128	85128	85132	85128	85128	-	224	290
R.400.1000.60	1478	684	140816	140816	140816	140816	140816	140816	42	3	4
R.500.100.1	248502	0	4	4	4	4	4	4	9760	165	1455
R.500.100.15	16775	2972	3543	4517	4628	5361	-	5724	-	-	-
R.500.100.30	6649	1670	8600	9665	9665	9665	9665	9665	-	2144	2073
R.500.100.60	1819	830	18240	18240	18240	18240	18240	18240	11	7	7
R.500.1000.1	248502	0	1316	1316	1316	1316	1316	1316	9383	1101	1425
R.500.1000.15	17866	2980	32950	42222	43134	50725	-	54049	-	-	-
R.500.1000.30	6360	1626	91272	98987	98987	98987	98987	98987	-	368	397
R.500.1000.60	1805	840	178212	178212	178212	178212	178212	178212	26	7	7
R.600.100.1	358202	0	1	1	1	1	1	1	6652	29	857
R.600.100.15	21474	3603	3656	4713	4803	5684	-	6254	-	-	-
R.600.100.30	7323	1990	11841	12465	12465	12465	12465	12465	-	610	640
R.600.100.60	1980	991	23293	23293	23293	23293	23293	23293	8	13	14
R.600.1000.1	358202	0	1337	1337	1337	1337	1337	1337	23005	246	1083
R.600.1000.15	23395	3778	36546	46293	47042	57237	-	61164	-	-	-
R.600.1000.30	7603	1923	116037	126798	126798	126798	126798	126798	-	2303	2050
R.600.1000.60	2196	1001	214608	214608	214608	214608	214608	214608	9	13	13
R.700.100.1	487902	0	1	1	1	1	1	1	13782	270	4842
R.700.100.15	25338	4334	4494	5845	5946	7311	-	7427	-	-	-
R.700.100.30	8606	2267	13663	14510	14510	14510	14510	14510	-	429	524
R.700.100.60	2514	1146	24102	24102	24102	24102	24102	24102	46	24	25
R.700.1000.1	487902	0	1231	1231	1231	1231	1231	1231	56712	1006	13341
R.700.1000.15	25845	4409	40662	53455	54351	66837	-	73997	-	-	-
R.700.1000.30	9104	2327	118718	134474	134474	134474	134474	134474	-	15865	7934
R.700.1000.60	2592	1194	245589	245589	245589	245589	245589	245589	75	24	24

precedence relations but for large graphs it was better to ignore them since the separation problem consumed too much time. We used the same primal heuristics as for the m -PDTSP which were designed to also consider capacities and thus lead to unnecessary long CPU times in some cases. However, heuristics for the SOP were not the aim of this paper.

In Table 7 we compare two variants of our branch-and-cut algorithms to the state-of-the-art results for the SOPLIB instances by Montemanni et al. (2013) which consist of 200 to 700 nodes. The existing approach (MO) had a CPU time limit of 2 days, whereas we set a limit of 1 day. For both branch-and-cut variants we set $\Delta_G = 0.9$, deactivate inequalities (24) and the heuristic separation. In BCH we run our primal heuristics whereas in BC we skip them to save time. We were able to solve 12 instances for the first time and significantly improved the lower bounds for the residual 12 open instances.

8. Conclusions

In this paper we have addressed the one-to-one multi-commodity pickup and delivery traveling salesman problem (m -PDTSP). We have shown that that the

m -PDTSP is equivalent to the 1-PDTSP (a different variant of pickup and delivery problems where only a single commodity is considered) with additional precedence constraints defined by the source-destination pairs of each commodity and have taken advantage of this relation to provide models for the m -PDTSP that are built by combining two different modeling components: one modeling flows and capacity constraints and the other modeling precedence relations. With respect to the precedence relation component, we have also introduced new inequalities based on sequences and logical implications of precedence relations which are able to significantly enhance the LP bounds, especially for instances with a large number of precedence constraints. For the capacity constraint component we have also presented alternative ways to model the capacity constraints based on load-dependent layered graphs which are beneficial for tight capacities in terms of LP bounds. Several variants of a branch-and cut algorithm were developed based on the presented models. These approaches were combined with several preprocessing methods, primal heuristics, and separation routines for the SOP inequalities. Especially for tightly capacitated instances with a large number of commodities we are able to outperform the approaches by Hernández-Pérez & Salazar-González (2009). Additionally, we have also considered the uncapacitated m -PDTSP which is equivalent to the TSP with precedence constraints (or sequential ordering problem). Here, an adapted variant of our branch-and-cut algorithm is able to solve to optimality several open instances from the TSPLIB and the SOPLIB.

References

- Abeledo, H., Fukasawa, R., Pessoa, A., & Uchoa, E. (2013). The time dependent traveling salesman problem: polyhedra and algorithm. *Mathematical Programming Computation*, 5, 27–55.
- Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). *Network flows: theory, algorithms, and applications*. Prentice Hall.
- Anghinolfi, D., Montemanni, R., Paolucci, M., & Gambardella, L. M. (2011). A hybrid particle swarm optimization approach for the sequential ordering problem. *Computers & Operations Research*, 38, 1076–1085.
- Ascheuer, N. (1995). *Hamiltonian path problems in the on-line optimization of flexible manufacturing systems*. Ph.D. thesis Technische Universität Berlin.
- Ascheuer, N., Jünger, M., & Reinelt, G. (2000). A branch & cut algorithm for the asymmetric traveling salesman problem with precedence constraints. *Computational Optimization and Applications*, 17, 61–84.
- Balas, E., Fischetti, M., & Pulleyblank, W. R. (1995). The precedence-constrained asymmetric traveling salesman polytope. *Mathematical Programming*, 68, 241–265.
- Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., & Laporte, G. (2007). Static pickup and delivery problems: a classification scheme and survey. *Top*, 15, 1–31.

- Cire, A. A., & Hoeve, W.-j. V. (2013). *Multivalued Decision Diagrams for Sequencing Problems*. Technical Report Tepper School of Business, Carnegie Mellon University.
- Dantzig, G., Fulkerson, R., & Johnson, S. (1954). Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America*, *2*, 393–410.
- Feo, T. A., & Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, *6*, 109–133.
- Gambardella, L. M., & Dorigo, M. (2000). An ant colony system hybridized with a new local search for the sequential ordering problem. *INFORMS Journal on Computing*, *12*, 237–255.
- Godinho, M. T., Gouveia, L., & Pesneau, P. (2011). On a time-dependent formulation and an updated classification of atsp formulations. In A. R. Mahjoub (Ed.), *Progress in Combinatorial Optimization* (pp. 223–254). ISTE-Wiley.
- Godinho, M. T., Gouveia, L., & Pesneau, P. (2014). Natural and extended formulations for the time-dependent traveling salesman problem. *Discrete Applied Mathematics*, *164*, 138–153.
- Gouveia, L., Leitner, M., & Ljubić, I. (2014a). Hop constrained steiner trees with multiple root nodes. *European Journal of Operational Research*, *236*, 100–112.
- Gouveia, L., Leitner, M., & Ljubić, I. (2014b). The two-level diameter constrained spanning tree problem. *Mathematical Programming*, (pp. 1–30). To appear.
- Gouveia, L., & Pesneau, P. (2006). On extended formulations for the precedence constrained asymmetric traveling salesman problem. *Networks*, *48*, 77–89.
- Gouveia, L., Simonetti, L. G., & Uchoa, E. (2011). Modeling hop-constrained and diameter-constrained minimum spanning tree problems as steiner tree problems over layered graphs. *Mathematical Programming*, *128*, 123–148.
- Gouveia, L., & Voß, S. (1995). A classification of formulations for the (time-dependent) traveling salesman problem. *European Journal of Operational Research*, *83*, 69–82.
- Hansen, P., & Mladenović, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, *130*, 449–467.
- Hernández-Pérez, H., & Salazar-González, J.-J. (2003). The one-commodity pickup-and-delivery travelling salesman problem. In M. Jünger, G. Reinelt, & G. Rinaldi (Eds.), *Combinatorial Optimization - Eureka, You Shrink!* (pp. 89–104). Berlin, Heidelberg: Springer volume 2570 of *LNCS*.
- Hernández-Pérez, H., & Salazar-González, J.-J. (2004). A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Applied Mathematics*, *145*, 126–139.

- Hernández-Pérez, H., & Salazar-González, J.-J. (2007). The one-commodity pickup-and-delivery traveling salesman problem: Inequalities and algorithms. *Networks*, *50*, 258–272.
- Hernández-Pérez, H., & Salazar-González, J.-J. (2009). The multi-commodity one-to-one pickup-and-delivery traveling salesman problem. *European Journal of Operational Research*, *196*, 987–995.
- Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., & Shmoys, D. B. (1985). *The traveling salesman problem: a guided tour of combinatorial optimization*. Wiley New York.
- Letchford, A. N., & Salazar-González, J.-J. (2006). Projection results for vehicle routing. *Mathematical Programming*, *105*, 251–274.
- Ljubić, I., & Gollowitzer, S. (2013). Layered graph approaches to the hop constrained connected facility location problem. *INFORMS Journal on Computing*, *25*, 256–270.
- Montemanni, R., Mojana, M., Di Caro, G. A., & Gambardella, L. M. (2013). A decomposition-based exact approach for the sequential ordering problem. *Journal of Applied Operational Research*, *5*, 2–13.
- Padberg, M., & Rinaldi, G. (1991). A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM review*, *33*, 60–100.
- Picard, J. C., & Queyranne, M. (1978). The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Operations Research*, *26*, 86–110.
- Roberti, R., & Toth, P. (2012). Models and algorithms for the asymmetric traveling salesman problem: an experimental comparison. *EURO Journal on Transportation and Logistics*, *1*, 113–133.
- Rodríguez-Martín, I., & Salazar-González, J. J. (2012). A hybrid heuristic approach for the multi-commodity one-to-one pickup-and-delivery traveling salesman problem. *Journal of Heuristics*, *18*, 849–867.
- Ruthmair, M., & Raidl, G. R. (2011). A layered graph model and an adaptive layers framework to solve delay-constrained minimum tree problems. In O. Günlük, & G. Woeginger (Eds.), *Proceedings of the 15th Conference on Integer Programming and Combinatorial Optimization (IPCO XV)* (pp. 376–388). Springer volume 6655 of *LNCS*.
- Uchoa, E. (2011). Cuts over extended formulations by flow discretization. In A. R. Mahjoub (Ed.), *Progress in Combinatorial Optimization* (pp. 255–282). ISTE-Wiley.