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

Luis Gouveia[a,1], Mario Ruthmair[b,c,2,*]

[a] *Centro de Investigacao Operacional & Faculdade de Ciencias, University of Lisbon, 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, 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 (sequential ordering problem) we are able to solve to optimality several open instances from the TSPLIB and SOPLIB.

Keywords: Transportation, Traveling Salesman, Sequential ordering

*Corresponding author

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

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

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

problem, Pickup and Delivery, Precedence constraints

1. Introduction

We address the one-to-one multi-commodity pickup and delivery traveling salesman problem (m-PDTSP) introduced by Hernández-Pérez & Salazar-González (2009). The problem arises in several transportation and logistics applications. We are given a complete directed graph with a node set consisting of start and end depot and a set of customers. For each arc a travel distance (or cost) is given. Furthermore, a set of commodities is given, each one associated with a demand, a source and a destination node. The capacity of the single vehicle is limited. The objective is to find a minimum-cost Hamiltonian path such that the vehicle starts and ends at the corresponding depot, the source of each commodity is visited before the associated destination, and the vehicle capacity is an upper bound of the vehicle load throughout the path satisfying all demands.

More formally, we are given a complete directed graph $G = (V, A)$. Node set V consists of start and end depot 0 and $n+1$, respectively, and the set of customers $V_c = \{1, ..., n\}$. For each arc $(i, j) \in A$, a travel distance (or cost) c_{ij} of going from i to j is given. There are m commodities $K = \{1, ..., m\}$, 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. The capacity of the vehicle is represented by $Q > 0$. We assume that $q_k \leq Q$ for all $k \in K$. The objective is to find a minimum cost sequence θ of nodes V such that: i) $\theta(0) = 0, \theta(n+1) = n+1$, i.e., the vehicle route starts and ends at the corresponding depot, ii) $\theta(s_k) < \theta(d_k), \forall k \in K$, i.e., the source is visited before the corresponding destination, and iii) $\sum_{k:\theta(s_k)\leq p<\theta(d_k)} q_k \leq Q, \forall p \in \{0, ..., n\}$, i.e., the vehicle capacity is an upper bound of the vehicle load for each position p on the path from 0 to $n+1$. The value $\theta(j)$ can be interpreted as the position of node $j \in V$ in the Hamiltonian path. The problem is $\mathcal{NP}$-hard since it generalizes the traveling salesman problem (TSP).

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 resulting 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. Our aim is to especially consider these type of instances and present models and solution algorithms to quickly solve them. 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 MIP neighborhood structures.

A different problem variant called the many-to-many one-commodity pickup and delivery traveling salesman problem (1-PDTSP) is introduced by Hernández-Pérez & Salazar-González (2003). In this problem we just consider 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 vehicle capacity Q. It is $\mathcal{NP}$-hard to find a feasible solution for the 1-PDTSP (Hernández-Pérez & Salazar-González, 2003) and since the 1-PDTSP is a relaxation of the m-PDTSP (Hernández-Pérez & Salazar-González, 2009), it is also $\mathcal{NP}$-hard to find a feasible solution for the m-PDTSP. Hernández-Pérez & Salazar-González (2004, 2007) present several models and valid inequalities for the 1-PDTSP and branch-and-cut algorithms to solve it. Dumitrescu et al. (2010) consider the TSP with pickup and delivery which is a special variant of the m-PDTSP without capacity constraints and a special commodity structure and introduce several new sets of valid inequalities and a branch-and-cut algorithm. An overview on further pickup and delivery problems can be found in Berbeglia et al. (2007).

We will show that the m-PDTSP is equivalent to the 1-PDTSP with additional precedence constraints defined by the origin-destination pairs for each commodity. The customer demands of the equivalent 1-PDTSP are defined by the load changes when the vehicle visits a customer in the m-PDTSP. 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. The precedence relations are ensured separately by adding inequalities for the sequential ordering polytope (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 gap, especially for instances with a large number of precedence constraints. Furthermore, we present alternative ways to model the capacity constraints based on load-dependent layered graphs which are beneficial for tight capacities in terms of LP bounds. In particular we consider a formulation based on a 3-dimensional layered graph that combines position and load together and even achieves tighter LP bounds, at the cost of a large model size. 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). Additionally, we consider the uncapacitated m-PDTSP which is equivalent to the TSP with precedence constraints (TSPPC) (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 SOPLIB.

The remainder of this article is 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 presents existing and new sets of valid inequalities, Section 7 describes our branch-and-cut algorithms, Section 8 shows experimental results, and Section 9 concludes the paper.

2. Preprocessing

In this section we discuss some problem reductions and important problem properties which will be used to reduce and strengthen the models discussed in this paper. Additionally, this information 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 $\tilde{R} \subseteq R$ 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 is different to 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 $V_k^{\text{in}} \subseteq V$ of nodes 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:

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

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:

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

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

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

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 valid preceding and succeeding arcs, respectively:

$$A^-_{ij} := \{(k, i) \in A : k \neq j, (j, k) \notin R, (k, l) \notin R \text{ for some } l \neq i, (l, j) \in R\}$$
$$A^+_{ij} := \{(j, k) \in A : k \neq i, (k, i) \notin R, (l, k) \notin R \text{ for some } l \neq j, (i, l) \in R\}$$

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:

$$l^-_{ij} = \sum_{k: A^-_{ij} \subseteq A^{\text{in}}_k \vee (i \neq s_k \wedge (i,j) \in A^{\text{in}}_k)} q_k, \qquad u^-_{ij} = \min\{u_{ij} - \rho_i, \max_{(k,i) \in A} u_{ki}\}$$
$$l^+_{ij} = \sum_{k: A^+_{ij} \subseteq A^{\text{in}}_k \vee (j \neq d_k \wedge (i,j) \in A^{\text{in}}_k)} q_k, \qquad u^+_{ij} = \min\{u_{ij} + \rho_j, \max_{(j,k) \in A} u_{jk}\}$$

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 already 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 (Hernández-Pérez & Salazar-González, 2009)

First, 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 and 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 these constraints, although exponential in number, can be easily implicitly included in the model by a cutting plane approach finding violated inequalities with max-flow computations (see, e.g., Ahuja et al., 1993).

We start by revising the flow model by Hernández-Pérez & Salazar-González (2009). This model is based on the generic scheme mentioned before. Flows and the vehicle capacity are ensured by adding for each commodity $k \in K$ and each arc (i,j), 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:

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

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

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

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. As mentioned by Hernández-Pérez & Salazar-González (2009) the LP relaxation of the model can be improved by adding the following well known modeling strengthening of multicommodity flow models extended by information obtained in preprocessing:

$$f_{ij}^k \begin{cases} = 0 & \text{if } (i,j) \in A_k^{\text{out}} \\ = q_k x_{ij} & \text{if } (i,j) \in A_k^{\text{in}} \\ \leq q_k x_{ij} & \text{else} \end{cases} \qquad \forall (i,j) \in A, \forall k \in K \tag{10}$$

$$l_{ij} x_{ij} \leq \sum_{k \in K} f_{ij}^k \leq u_{ij} x_{ij} \qquad \forall (i,j) \in A \tag{11}$$

Sets A_k^{in} and A_k^{out} for each $k \in K$ and load bounds l_{ij} and u_{ij} are defined in Section 2.3.

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, this "equivalence" relation has never been stated neither used before. A related relation has been given 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 system (7)–(9) described in the previous section into a different and equivalent system where this relation is enhanced.

4.1. Introducing Scaled Flow Variables

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

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

to rewrite (7) and (9) as follows:

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

$$g_{ij}^k \geq 0 \qquad \forall (i,j) \in A, \forall k \in K \tag{14}$$

4.2. Aggregating the Flows

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

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

The right-hand side of (15) corresponds to the already defined net demand values $\rho_i, \forall i \in V$. By using aggregated flow variables $f_{ij}, \forall (i,j) \in A$, and equalities

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

we can rewrite (8), (9) and (15) as the following single-commodity flow (SCF) system:

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

$$0 \leq f_{ij} \leq Q x_{ij} \qquad \forall (i,j) \in A \qquad (18)$$

$$f_{ij} \geq 0 \qquad \forall (i,j) \in A \qquad (19)$$

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

It is now easy to see that in terms of integer solutions, the aggregated flow system (17)–(19) together with (13)–(14), the linking constraints

$$f_{ij} = \sum_{k\in K} q_k g_{ij}^k \qquad \forall (i,j) \in A, \qquad (20)$$

and (6), is equivalent to the model (7)–(9) and (6).

In terms of linear programming relaxations, the equivalence is also obvious. One direction has already been proved with the given transformation. To see the reverse situation, note that from a given solution feasible for the system defined by (13)–(14), (17)–(19), and (20), we obtain a feasible solution for (7)–(9) simply by setting the f_{ij}^k variables as defined by (12).

Thus, we have just proved that:

Result 4.1. *Under the transformation* (12) *the system defined by* (13)–(14), (17)–(19), *and* (20), *is equivalent to the system* (7)–(9).

A similar result can be obtained by adding the strengthening inequalities (10)–(11) to the system (7)–(9), and equivalently constraints

$$g_{ij}^k \begin{cases} = 0 & \text{if } (i,j) \in A_k^{\text{out}} \\ = x_{ij} & \text{if } (i,j) \in A_k^{\text{in}} \\ \leq x_{ij} & \text{else} \end{cases} \qquad \forall (i,j) \in A, \forall k \in K \qquad (21)$$

$$l_{ij} x_{ij} \leq f_{ij} \leq u_{ij} x_{ij} \qquad \forall (i,j) \in A \qquad (22)$$

to system (13)–(14), (17)–(19), and (20).

Table 1: Model TMCF

$(1)-(4), (6)$		
$f(\delta^+(i)) - f(\delta^-(i))$	$= \rho_i$	$\forall i \in V$
$l_{ij}x_{ij} \leq f_{ij}$	$\leq u_{ij}x_{ij}$	$\forall (i,j) \in A$
$g^k(\delta^+(i)) - g^k(\delta^-(i))$	$= \begin{cases} 1 & \text{if } i = s_k \\ -1 & \text{if } i = d_k \\ 0 & \text{else} \end{cases}$	$\forall i \in V \setminus V_k^{\text{out}}, \forall k \in K$
$0 \leq g^k_{ij}$	$\begin{cases} = 0 & \text{if } (i,j) \in A_k^{\text{out}} \\ = x_{ij} & \text{if } (i,j) \in A_k^{\text{in}} \\ \leq x_{ij} & \text{else} \end{cases}$	$\forall (i,j) \in A, \forall k \in K$
f_{ij}	$= \sum_{k \in K} q_k g^k_{ij}$	$\forall (i,j) \in A$

Result 4.2. *Under the transformation* (12) *the system defined by* (13)–(14), (21), *and* (17), (22), *and* (20) *is equivalent to the system* (7), (9), (10) *and* (11).

We denote by MCF the original model from Hernández-Pérez & Salazar-González (2009) with constraints (10)–(11) and by "Transformed MCF" (TMCF) the model just derived including constraints (21)–(22). Here, we refer to the complete models for the whole problem. Results 4.1 and 4.2 state that the two models provide the same linear programming bound.

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

In order to motivate this relation, consider Table 1 that gives an overview of the essential parts in model TMCF.

In order to make the connection that we have mentioned at the beginning of this section, we first remove the linking constraints (20) from the TMCF model (at the end of Table 1). We denote by Weak TMCF (WTMCF) the model obtained in this way. We observe that WTMCF is still a valid model for the problem, although with a weaker LP relaxation.

Theorem 4.3. *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) and (22) 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) and (22) (see Hernández-Pérez & Salazar-González, 2004).

Inductive step: We assume that model WTMCF is valid for the $(m-1)$-PDTSP with commodities $K = \{1, ..., m-1\}$. We want to show that model WTMCF stays valid when adding a further commodity m. The additional flow system (13)–(14), (21), 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) and (22) for nodes $i = s_m, d_m$ ensure that the additional demand q_m is considered with respect to the vehicle load bounds. □

Second, we observe that the aggregated flow system on variables f_{ij} (see second box in Table 1) ensures the capacity constraints as well as the net demands and corresponds to the flow system in formulations for the 1-PDTSP (e.g., Hernández-Pérez & Salazar-González, 2004, 2007). Finally, the g^k_{ij} system (see third 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) and (21).

4.5. Modeling the Precedence Constraints with SOP Inequalities

We present next one alternative for modeling the precedence constraints using cut-like inequalities, the so-called sequential ordering polytope (SOP) inequalities, known from the literature to guarantee precedence constraints (Balas et al., 1995; Ascheuer et al., 2000). For each commodity $k \in K$ we define a set of relevant nodes $V^k = V \backslash V_k^{\text{out}}$ and the corresponding inequalities are defined as follows:

$$x(S, V^k \setminus S) \geq 1 \qquad \forall S \subset V^k, s_k \in S, d_k \in V^k \setminus S, \forall k \in K \tag{23}$$

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.

We denote model WTMCF with the flow system (13)–(14), (21) replaced by SOP cuts (23) by CUTK. Inequalities (23) 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. Also, as a 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), (21) 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 (23) and $0 \leq x_{ij} \leq 1$, that is:

Result 4.4. $Proj_x(F(WTMCF_L)) = Proj_x(F(CUTK_L))$.

As a consequence of this result, the bounds obtained from the LP relaxations of model CUTK and WTMCF are the same.

As pointed out before, 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 notorious for cases with tight capacity. However, in instances with too many commodities, this alternative view may be preferable (which is confirmed by our computational results) to the one of including a flow system associated to each commodity as with the TMCF model.

4.6. Strengthening the Cut Model

Additionally, we can strengthen the model by adding other families of precedence related cut-like inequalities to the model. Besides considering the source-target pairs (s_k, d_k) for each commodity $k \in K$ to define associated SOP inequalities, other sets of node pairs, i.e., all non-transitive precedence relations $\tilde{R}$, will be used. Essentially, the additional node pairs relate depot nodes 0 and $n+1$ to other nodes, i.e.,

$$\begin{aligned} \tilde{R} = & \{(s_k, d_k) : k \in K\} \\ & \cup \{(0, i) : i \in V \setminus \{0, n+1\}, (j, i) \notin R, \forall j \in V \setminus \{0\}\} \\ & \cup \{(i, n+1) : i \in V \setminus \{0, n+1\}, (i, j) \notin R, \forall j \in V \setminus \{n+1\}\} \end{aligned} \quad (24)$$

Similar to the inequalities (23) 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:

$$x(S, V^{ij} \setminus S) \geq 1 \qquad \forall S \subset V^{ij}, i \in S, j \in V^{ij} \setminus S, \forall (i, j) \in \tilde{R} \quad (25)$$

If $i = 0$, inequalities (35) are known as weak σ-inequalities, if $j = n+1$ as weak π-inequalities, and if $i \neq 0$ and $j \neq n+1$ as simple (π, σ)-inequalities (Balas et al., 1995). 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 (23) replaced by (25) by CUTR. 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.

Figure 1 summarizes the strength relations of the LP relaxations of the discussed models. We note that there is no LP relation between the bounds given by the models MCF and CUTR. This can be observed for the experimental tests, e.g., in Table 2.

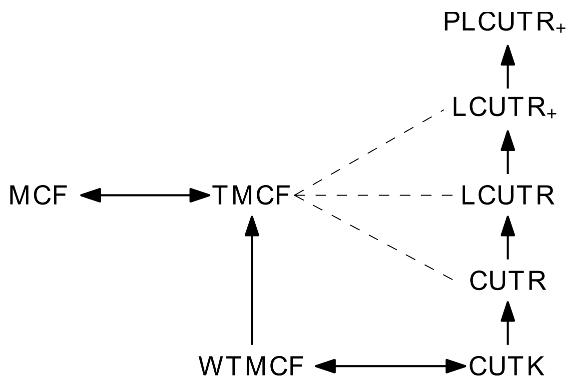


Figure 1: Strength relations: The model at the head of an arrow provides an optimal LP relaxation value which is at least as good as the one of the model at the corresponding tail. Dashed lines indicate that there is no LP relation between the two models.

5. 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, 2014a,b; 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

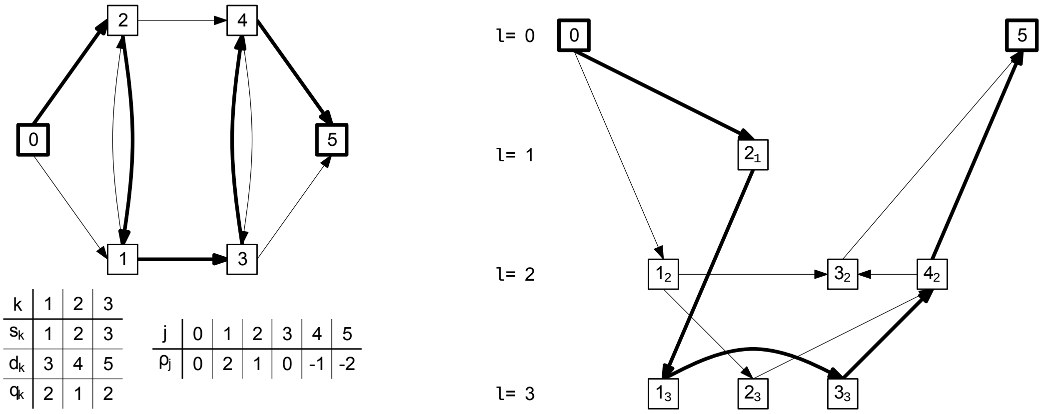
In this subsection we show that the model by Picard & Queyranne (1978) (PQ) can be easily readapted to model the capacity constraints of the aggregated SCF model (17) and (22). We consider the variables z^l_{ij} for each arc $(i, j) \in A$ and each possible vehicle load $l \in L_{ij} := \{l_{ij}, ..., u_{ij}\}$. 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 model is defined as follows:

$$z^{l-\rho_j}(\delta^-(j)) = z^l(\delta^+(j)) \qquad \forall j \in V_c, \forall l \in L_j \tag{26}$$

$$\sum_{l\in L_{ij}} z^l_{ij} = x_{ij} \qquad \forall (i, j) \in A \tag{27}$$

$$z^l_{ij} \geq 0 \qquad \forall (i, j) \in A, \forall l \in L_{ij} \tag{28}$$

We denote the model CUTR in which the SCF system (17), (22) is replaced by system (26)–(28) by LCUTR. 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



k	1	2	3
s_k	1	2	3
d_k	3	4	5
q_k	2	1	2

j	0	1	2	3	4	5
ρ_j	0	2	1	0	-1	-2

Figure 2: In the left a problem instance with graph G with $\mathsf{n} = 4$, 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 $\mathsf{Q} = 3$ is shown. The set of bold arcs in both graphs represent the same feasible solution.

and in fact the experimental results showed that for many instances it is better. Note that in contrast to the original time-dependent PQ model the load-dependent PQ model alone is not sufficient to eliminate subtours since values ρ_j may also be negative. However, in the model CUTR as well as in LCUTR subtour elimination is guaranteed by SOP cuts (25) (which dominate connection cuts (4)).

5.2. Strengthening the Load-Dependent PQ Model

We can view the load-dependent PQ system (26)–(28) as modeling a path in a layered graph $G_\mathrm{L} = (V_\mathrm{L}, A_\mathrm{L})$. 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 $V_\mathrm{L} = \{0, n+1\} \cup \{j_l : j \in V_c, l \in L_j\}$ 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.2.

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 (25) in the load-based layered graph G_{L} to improve the LP relaxation of model LCUTR. Let $S_{\mathrm{L}} := \{j_l \in V_{\mathrm{L}} : j \in S\}$ denote the set of all copies of nodes in some set $S \subseteq V$. The corresponding SOP cuts in G_{L} are defined as:

$$z(S, V_{\mathrm{L}}^{ij} \setminus S) \geq 1 \quad \forall S \subset V_{\mathrm{L}}^{ij}, \{i\}_{\mathrm{L}} \subseteq S, \{j\}_{\mathrm{L}} \subseteq V_{\mathrm{L}}^{ij} \setminus S, \forall (i,j) \in \tilde{R} \quad (29)$$

These inequalities can be interpreted as the SOP cuts (25) lifted in the load layered graph G_{L}. It is easy to see that SOP cuts (25) are implied by (29) since the subset of (29) in which all copies of nodes $v \in V^{ij} \setminus \{i,j\}$ belong either to S or to $V_{\mathrm{L}}^{ij} \setminus S$ gives exactly the SOP cuts (25) for a precedence relation $(i,j) \in \tilde{R}$. We denote the model LCUTR replacing SOP cuts (25) by (29) by LCUTR$_+$. 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, there is no LP relation between MCF and LCUTR$_+$, as can be observed e.g., in Table 2.

5.3. The Position-Load-Dependent PQ Model

As noted before, the layered graph associated to the load-dependent PQ model is not acyclic. However, the good results taken from Godinho et al. (2011, 2014) explicitly use the fact that the associated layered graphs are acyclic (as in the original PQ model) 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 $P_j := \{\alpha_j, ..., \omega_j\}$ be the set of possible positions for node $j \in V$. We also define the set of possible positions $P_{ij} := \{\max\{\alpha_i + 1, \alpha_j\}, ..., \min\{\omega_i + 1, \omega_j\}\}$ 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 model is defined as:

$$z^{p,l-\rho_j}(\delta^-(j)) = z^{p+1,l}(\delta^+(j)) \qquad \forall j \in V_c, \forall p \in P_j, \forall l \in L_j \qquad (30)$$

$$\sum_{p \in P_{ij}} \sum_{l \in L_{ij}} z_{ij}^{pl} = x_{ij} \qquad \forall (i,j) \in A \qquad (31)$$

$$z_{ij}^{pl} \geq 0 \qquad \forall (i,j) \in A, \forall p \in P_{ij}, \forall l \in L_{ij} \qquad (32)$$

Again, 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 model (30)–(32) is sufficient to eliminate subtours.

Similarly to what has been suggested in the last subsection we readapt the SOP cuts (25) 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:

$$z(S, V_{\text{PL}}^{ij} \setminus S) \geq 1 \quad \forall S \subset V_{\text{PL}}^{ij}, \{i\}_{\text{PL}} \subseteq S, \{j\}_{\text{PL}} \subseteq V_{\text{PL}}^{ij} \setminus S, \forall (i,j) \in \tilde{R} \quad (33)$$

These inequalities can be interpreted as the SOP cuts (25) 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 (33) dominate the SOP cuts (29) in the load layered graph G_{L}. We denote the generic model extended by (30)–(32), and (33) by PLCUTR$_+$. 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 LP relation to MCF is still open. Finally, Figure 1 summarizes the strength relations of all models proposed in the last two sections.

6. Valid Inequalities

In this section we consider other sets of valid inequalities that are based on exploiting both the precedence relations and capacity constraints to strengthen the LP relaxation of the models discussed in the previous sections. Some of the precedence based inequalities are taken from the literature. However, we also create new generalizations that are useful in practice and are based on exploiting sequences of precedence pairs.

6.1. Precedence-Based Inequalities in Original Graph

We adopt the notation from Balas et al. (1995) and write $\pi(S) := \{i : (i, j) \in R, j \in S\}$ for the set of predecessors for some subset $S \subseteq V$. Similarly, we use $\sigma(S) := \{j : (i, j) \in R, i \in S\}$ to denote the corresponding set of successors. If $S = \{i\}$ we simply write $\pi(i)$ and $\sigma(i)$, respectively. The π-, σ-, and (π, σ)-inequalities have been proposed for the TSPPC by Balas et al. (1995) and are defined as:

$$x(S \setminus \pi(S), V \setminus (S \cup \pi(S))) \geq 1 \qquad \forall S \subset V \setminus \{n+1\} \qquad (34)$$

$$x(V \setminus (S \cup \sigma(S)), S \setminus \sigma(S)) \geq 1 \qquad \forall S \subset V \setminus \{0\} \qquad (35)$$

$$x(S \setminus S', V \setminus (S \cup S')) \geq 1 \qquad \forall S \subset V, S' = \pi(X) \cup \sigma(Y), \forall X, Y \subset V_c, X \subseteq S, Y \subseteq V \setminus S, \text{with } (i, j) \in R, \forall i \in X, j \in Y \qquad (36)$$

For any violated SOP cut (25) we check if it can be lifted to a corresponding inequality (34)–(36). This can be easily done for inequalities (34) and (35). On the other hand, it is not obvious how to choose sets X and Y for the (π, σ)-inequalities (36) for a given violated inequality (25) with set S and nodes i, j. We consider two different cases for the liftings: i) $X = \{i\}, Y = \{v : (i, v) \in R, v \in V \setminus S\}$, and ii) $X = \{v : (v, j) \in R, v \in S\}, Y = \{j\}$.

A subset of the precedence cycle breaking inequalities (PCB) by Balas et al. (1995) is defined as follows: Let $S \subset V \setminus \{n+1\}$ and $i_1, i_3 \in S, i_2 \notin S$, with $(i_1, i_2), (i_2, i_3) \in R$. Then,

$$x(S, V \setminus S) \geq 2. \qquad (37)$$

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 inequalities (37) by considering sequences of precedence relations $(i_1, i_2), ..., (i_{k-1}, i_k) \in R, i_1, ..., i_k \in V \setminus \{n+1\}$, for odd values of $k \geq 3$. We require all odd nodes to be in set $S \subset V \setminus \{n+1\}$ and all even nodes to be in set $V \setminus S$, i.e., $\{i_h : h \leq k, h \text{ odd}\} \subseteq S$ and

$\{i_h : h \leq k, h \text{ even}\} \subset V \setminus S$. 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

$$x(S, V \setminus S) \geq \lceil k/2 \rceil. \tag{38}$$

Note that due to the right-hand side of (38) the inequalities for sequences with even k are dominated by the ones for the corresponding sequence where the last node is removed. 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 $(i = i_1, i_2, ..., i_k = j)$ 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 (38).

Inequalities (37) and (38) 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 $h = 1, ..., k$, are assigned to the sets as defined above.

Since inequalities (37) and (38) consider the path starting from i_1, passing all nodes $i_h, h = 2, ..., k$, 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 must not directly follow 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

$$x(S \setminus S', V \setminus (S \cup S')) \geq \lceil k/2 \rceil. \tag{39}$$

Similar to the SOP cuts (38), inequalities (39) 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 next set of inequalities are based on logical implications to fix variables in a branch-and-bound node (Ascheuer et al., 2000): 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:

$$x(\{i,j\}) + x(\{k,l\}) \le 1 \qquad \forall i,j \in V_c, i \neq j, \forall k \in \pi(i), \forall l \in \sigma(j) \qquad (40)$$
$$x(\{i,j\}) + x(k,\sigma(j)) \le 1 \qquad \forall i,j \in V_c, i \neq j, \forall k \in \pi(i) \qquad (41)$$
$$x(\{i,j\}) + x(\sigma(j),k) \le 1 \qquad \forall i,j \in V_c, i \neq j, \forall k \in \pi(i) \qquad (42)$$
$$x(\{i,j\}) + x(\pi(i),l) \le 1 \qquad \forall i,j \in V_c, i \neq j, \forall l \in \sigma(j) \qquad (43)$$
$$x(\{i,j\}) + x(l,\pi(i)) \le 1 \qquad \forall i,j \in V_c, i \neq j, \forall l \in \sigma(j) \qquad (44)$$

The validity of these inequalities can be easily shown by using node degree constraints (2) and (3). We add violated inequalities (40)–(44) within a cutting plane algorithm by examining them one by one.

6.2. Precedence-Based Inequalities in Layered Graphs

We consider similar valid inequalities in the variable space defined by the layered graphs G_{L} and G_{PL}, as it was done before in Section 5. 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 j since one or both copies may not be visited at all. Thus, the predecessors and successors of some subset $S \subseteq V_{\mathrm{L}}$ in G_{L} need to be defined as $\pi_{\mathrm{L}}(S) := \{i_l \in V_{\mathrm{L}} : (i,j) \in R \wedge \{j\}_{\mathrm{L}} \subseteq S\}$ and $\sigma_{\mathrm{L}}(S) := \{j_l \in V_{\mathrm{L}} : (i,j) \in R \wedge \{i\}_{\mathrm{L}} \subseteq S\}$, respectively. The definitions in G_{PL} are straightforward.

As already shown in Section 5 SOP inequalities (29) and (33) are lifted variants of SOP inequalities (25) in the layered graph. In a similar way we lift the π-, σ-, and (π,σ)-inequalities (34)–(36) to the space of variables z^l_{ij} and z^{pl}_{ij}, respectively. Here, we only mention the lifted inequalities in G_{L} since it is straightforward to formulate the corresponding inequalities in G_{PL}:

$$z(S \setminus \pi_{\mathrm{L}}(S), V \setminus (S \cup \pi_{\mathrm{L}}(S))) \ge 1 \qquad \forall S \subset V_{\mathrm{L}} \setminus \{n+1\} \qquad (45)$$
$$z(V \setminus (S \cup \sigma_{\mathrm{L}}(S)), S \setminus \sigma_{\mathrm{L}}(S)) \ge 1 \qquad \forall S \subset V_{\mathrm{L}} \setminus \{0\} \qquad (46)$$
$$z(S \setminus S', V \setminus (S \cup S')) \ge 1 \qquad \forall S \subset V_{\mathrm{L}}, S' = \pi_{\mathrm{L}}(X_{\mathrm{L}}) \cup \sigma_{\mathrm{L}}(Y_{\mathrm{L}}), \forall X,Y \subset V_c, X_{\mathrm{L}} \subseteq S, Y_{\mathrm{L}} \subseteq V_{\mathrm{L}} \setminus S, \text{ with } (i,j) \in R, \forall i \in X, j \in Y \qquad (47)$$

When a violated inequality (29) is found we try to lift it to the corresponding inequality (45)–(47). As mentioned above, in this case we need to take care of the different meaning of predecessors and successors.

Finally, we also lift inequalities (39) to the space of variables z^l_{ij}. Let $(i_1, i_2), ..., (i_{k-1}, i_k) \in R, i_1, ..., i_k \in V \setminus \{n+1\}$, for odd values of $k \geq 3$ be a non-dominated sequence of precedence relations as defined in Subsection 6.1. All copies of odd nodes in G_L are fixed to some set $S \subset V_\text{L} \setminus \{n+1\}$ and all copies of even nodes to set $V_\text{L} \setminus S$, i.e., $\bigcup_{h \leq k, h \text{ odd}} \{i_h\}_\text{L} \subseteq S$ and $\bigcup_{h \leq k, h \text{ even}} \{i_h\}_\text{L} \subset V_\text{L} \setminus S$. The set of excluded nodes is defined as $S' = \bigcup_{(j,i_1) \in R \vee (i_k, j) \in R \setminus \tilde{R}} \{j\}_\text{L}$. Then, we obtain inequalities

$$z(S \setminus S', V_\text{L} \setminus (S \cup S')) \geq \lceil k/2 \rceil. \tag{48}$$

Similar to inequalities (39) this lifted variant can be separated in polynomial time by computing the max-flow in a support graph $G''_\text{L} = (V'_\text{L} \setminus S', A'_\text{L} \setminus \{(i,j) : i \in S' \vee j \in S'\})$ with $V'_\text{L} = V_\text{L} \cup \{s,t\}$, and $A' = A_\text{L} \cup \{(s, i_h) : h \leq k, h \text{ odd}\} \cup \{(i_h, t) : h \leq k, h \text{ even}\}$. 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.

6.3. Capacity-Based Inequalities in Original Graph

In case of a violated inequality (4), (37), and (38) for some set S we check if the corresponding rounded capacity cut (Letchford & Salazar-González, 2005) in the context of the m-PDTSP is stronger:

$$x(\delta^+(S)) \geq \left\lceil \frac{\sum_{k: s_k \in S \wedge d_k \notin S} q_k}{\max_{(i,j) \in \delta^+(S)} u_{ij}} \right\rceil \tag{49}$$

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

7. 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. 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.

7.1. 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.

7.2. 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 $x^{\mathrm{LP}}(A') < \Delta_G \cdot b$ with $\Delta_G \in (0, 1]$. Similarly, we use parameters $\Delta_{G_{\mathrm{L}}}$ and $\Delta_{G_{\mathrm{PL}}}$ for valid inequalities in G_{L} and G_{PL}, respectively.
- If the LP relaxation value did 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.

Next to 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 inequalities (34)–(36), and (49).

7.3. 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.

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 $c'_{ij} = c_{ij}(1 - x_{ij}^{\mathrm{LP}})$ 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 memorize 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 are increased by one (up to at most 10).

8. Experiments

This section shows and discusses experimental results for instances of the m-PDTSP and the TSPPC (or sequential ordering problem). 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.

8.1. 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 TSPPC, each precedence relation corresponding to a commodity with demand 1 (Suffix "max1") or with a randomly chosen demand in $\{1, ..., 5\}$ (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,

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

				LP gap in %										LP time in seconds									
				HS		CUT-			LCUT-			PLCUT-		HS		CUT-			LCUT-			PLCUT-	
Instance	$\|V\|$	$\|K\|$	Q	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
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
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	1	1
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	1	1
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	2	2
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	1	1	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	2	2	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	1	1	1	146	139
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	2	2	3	931	922
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	4	4	5	1582	1585
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
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
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
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
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
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	2	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	2	2	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	8	0	tl	57
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	28	75	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	1	1082	1	tl	93
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	3196	3952
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	2	3	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	1	0	tl	18
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	50	60	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	1	1264	4741	tl	tl
p43.1Q2max1	44	9	2	-	-	-	-	-	-	-	-	-	-	-	37	2	2	2	4	220	751	tl	tl
p43.1Q3max1	44	9	3	-	-	-	-	-	-	-	-	-	-	-	40	2	2	2	3	1754	1218	tl	tl
p43.1Q4max1	44	9	4	-	-	-	-	-	-	-	-	-	-	-	60	2	1	3	8	1544	2286	tl	tl
p43.1Q10max5	44	9	10	-	-	-	-	-	-	-	-	-	-	-	62	1	2	2	12	tl	tl	tl	tl
p43.1Q15max5	44	9	15	-	-	-	-	-	-	-	-	-	-	-	29	1	2	3	55	tl	tl	tl	tl
p43.2Q10max1	44	20	10	-	-	-	-	-	-	-	-	-	-	-	1154	3	3	4	64	tl	tl	tl	tl
p43.2Q40max5	44	20	40	-	-	-	-	-	-	-	-	-	-	-	223	2	2	4	964	tl	tl	tl	tl
p43.3Q10max1	44	37	10	-	-	-	-	-	-	-	-	-	-	-	2837	4	7	8	63	tl	tl	tl	tl
p43.3Q40max5	44	37	40	-	-	-	-	-	-	-	-	-	-	-	1006	3	5	8	1750	tl	tl	tl	tl
p43.4Q10max1	44	50	10	-	-	-	-	-	-	-	-	-	-	-	34	1	1	3	15	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

and demand in $\{1, ..., 5\}$. 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. 1. Additionally, we enhance model CUTR by considering all valid inequalities described in Section 6.1 and 6.3, and denote it by CUTR*. Similarly, we denote model LCUTR$_+$ with all valid inequalities in Section 6 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 7.2. Because of this and the heuristic liftings the LP relations to the models with exact deterministic separation are not consistent. Similarly, the Benders decomposition

Table 3: Comparison of LP relaxations of different models for class 2 and 3 instances. Bold values denote the best LP gaps. Each set contains 10 instances.

				avg. LP gap in %										avg. LP time in seconds									
				HS		CUT-			LCUT-			PLCUT-		HS		CUT-			LCUT-			PLCUT-	
Set	$\|V\|$	$\|K\|$	Q	**BE**	**MCF**	**K**	**R**	$\mathbf{R}^*$	**R**	$\mathbf{R}_+$	$\mathbf{R}^*_+$	$\mathbf{R}_+$	$\mathbf{R}^*_+$	**BE**	**MCF**	**K**	**R**	$\mathbf{R}^*$	**R**	$\mathbf{R}_+$	$\mathbf{R}^*_+$	$\mathbf{R}_+$	$\mathbf{R}^*_+$
n10m5Q10	11	5	10	1.7	1.7	1.9	1.6	0.5	0.6	**0.0**	**0.0**	**0.0**	**0.0**	0	0	0	0	0	0	0	0	0	0
n10m5Q15	11	5	15	1.2	0.6	0.6	0.5	**0.0**	0.2	**0.0**	**0.0**	**0.0**	**0.0**	0	0	0	0	0	0	0	0	0	0
n10m5Q20	11	5	20	-	0.6	0.6	0.5	**0.0**	0.2	**0.0**	**0.0**	**0.0**	**0.0**	-	0	0	0	0	0	0	0	0	0
n10m10Q10	11	10	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
n10m10Q15	11	10	15	3.5	0.4	1.6	1.6	1.5	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0	0	0	0	0	0	0	0	0	0
n10m10Q20	11	10	20	1.0	1.2	1.4	1.4	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0	0	0	0	0	0	0	0	0	0
n10m10Q25	11	10	25	-	**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
n10m10Q30	11	10	30	-	**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
n10m15Q10	11	15	10	-	-	-	-	-	-	-	-	-	-	-	inf	inf	inf	inf	inf	inf	inf	inf	inf
n10m15Q15	11	15	15	**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
n10m15Q20	11	15	20	**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
n10m15Q25	11	15	25	**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
n10m15Q30	11	15	30	**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
n15m5Q10	16	5	10	4.7	7.6	8.3	5.1	3.1	4.3	1.3	1.1	**0.0**	**0.0**	0	0	0	0	0	0	4	4	782	688
n15m5Q15	16	5	15	4.0	6.3	6.3	3.2	1.9	2.8	0.6	0.5	**0.1**	**0.1**	0	0	0	0	0	0	15	35	735	732
n15m5Q20	16	5	20	-	6.3	6.3	3.2	1.9	2.8	0.6	0.5	**0.1**	**0.1**	-	0	0	0	0	0	22	46	739	734
n15m10Q10	16	10	10	9.3	9.6	14.7	10.4	6.6	9.6	3.6	2.7	**0.0**	**0.0**	0	0	0	0	0	0	5	8	549	670
n15m10Q15	16	10	15	6.1	5.9	7.1	4.8	2.7	4.2	0.1	**0.0**	**0.0**	**0.0**	0	0	0	0	0	0	41	39	44	46
n15m10Q20	16	10	20	5.6	5.6	6.2	3.5	1.8	2.8	0.1	**0.0**	**0.0**	**0.0**	0	0	0	0	0	0	97	146	83	72
n15m10Q25	16	10	25	4.8	5.3	5.4	2.4	1.2	2.3	0.1	**0.0**	**0.0**	**0.0**	0	0	0	0	0	1	230	144	115	111
n15m10Q30	16	10	30	-	5.3	5.3	2.3	1.2	2.2	0.1	**0.0**	**0.0**	**0.0**	-	0	0	0	0	1	251	202	142	103
n15m15Q10	16	15	10	-	-	-	-	-	-	-	-	-	-	-	0	0	0	0	0	inf	inf	inf	inf
n15m15Q15	16	15	15	13.4	11.1	12.6	11.6	9.8	9.9	0.1	**0.0**	**0.0**	**0.0**	0	0	0	0	0	0	1	5	4	4
n15m15Q20	16	15	20	4.3	2.3	3.8	3.3	1.2	1.8	**0.0**	**0.0**	**0.0**	**0.0**	0	0	0	0	0	0	0	0	4	4
n15m15Q25	16	15	25	1.7	0.1	0.4	0.1	**0.0**	0.1	**0.0**	**0.0**	**0.0**	**0.0**	0	0	0	0	0	0	0	0	4	4
n15m15Q30	16	15	30	1.7	0.1	0.4	0.1	**0.0**	0.1	**0.0**	**0.0**	**0.0**	**0.0**	0	0	0	0	0	0	0	0	5	4
n20m5Q10	21	5	10	2.6	5.3	5.7	4.0	1.9	3.1	0.2	0.2	**0.0**	**0.0**	0	0	0	0	0	0	1	1	207	117
n20m5Q15	21	5	15	2.0	3.5	3.5	1.3	0.4	1.2	**0.0**	**0.0**	**0.0**	**0.0**	0	0	0	0	0	1	1	1	219	126
n20m5Q20	21	5	20	-	3.5	3.5	1.3	0.4	1.2	**0.0**	**0.0**	**0.0**	**0.0**	-	0	0	0	0	1	1	1	247	130
n20m10Q10	21	10	10	6.4	9.3	14.1	10.0	5.8	8.6	2.9	**2.0**	3.8	2.4	0	1	0	0	0	0	127	119	5453	4926
n20m10Q15	21	10	15	7.0	11.0	13.0	8.7	5.9	7.9	3.2	**2.5**	3.8	2.7	0	1	0	0	0	1	1000	948	5102	4431
n20m10Q20	21	10	20	3.7	8.6	8.8	4.3	1.8	4.2	1.3	**0.6**	1.7	1.0	0	0	0	0	0	2	1672	1672	2707	2440
n20m10Q25	21	10	25	3.4	8.1	8.1	3.5	1.5	3.5	1.1	**0.5**	1.5	0.8	0	0	0	0	0	3	1511	1274	2822	1988
n20m10Q30	21	10	30	-	8.1	8.1	3.5	1.5	3.5	1.1	**0.5**	1.5	0.8	-	0	0	0	0	3	1555	1475	2881	2103
n20m15Q10	21	15	10	-	-	-	-	-	-	-	-	-	-	-	2	0	0	0	0	25	60	tl	tl
n20m15Q15	21	15	15	13.4	17.1	20.5	17.9	13.6	16.3	6.8	5.5	6.4	**5.2**	0	2	0	0	0	1	1195	1248	tl	tl
n20m15Q20	21	15	20	11.3	15.2	17.0	13.8	8.5	13.3	3.9	**2.7**	6.1	4.2	0	1	0	0	0	2	4886	4677	6585	6430
n20m15Q25	21	15	25	12.2	11.5	12.2	8.9	4.5	8.6	2.6	**1.7**	4.2	2.5	0	1	0	0	0	3	4334	3259	5096	4519
n20m15Q30	21	15	30	11.2	10.6	11.1	7.7	3.3	7.4	2.2	**1.1**	3.6	1.7	0	1	0	0	0	5	4381	3499	5767	4333
n25m5Q10	26	5	10	3.6	6.7	7.2	5.4	3.7	4.6	1.6	**1.3**	2.1	1.8	0	1	0	0	0	1	35	37	4605	3519
n25m5Q15	26	5	15	3.5	4.7	4.8	3.0	2.1	2.8	0.7	**0.6**	1.1	0.9	0	0	0	0	0	3	120	167	4339	4269
n25m5Q20	26	5	20	3.2	4.5	4.5	2.7	1.9	2.6	0.7	**0.6**	1.1	0.9	0	0	0	0	0	3	211	234	4172	4077
n25m10Q10	26	10	10	10.0	11.9	17.7	14.5	10.2	13.7	7.6	**5.6**	10.3	7.6	0	3	0	0	0	1	218	171	tl	tl
n25m10Q15	26	10	15	9.1	12.5	14.0	10.9	7.6	10.6	4.7	**3.3**	8.4	6.0	0	1	0	0	0	3	2849	2656	tl	tl
n25m10Q20	26	10	20	7.5	10.9	11.2	7.9	5.0	7.7	3.2	**1.7**	6.8	4.5	0	1	0	0	0	5	5682	5109	6801	6707
n25m10Q25	26	10	25	7.2	10.5	10.5	7.3	4.5	7.2	3.3	**1.8**	6.7	4.7	0	1	0	0	0	8	6480	5107	6651	6736
n25m10Q30	26	10	30	-	10.5	10.5	7.3	4.5	7.2	3.5	**1.8**	7.1	4.9	-	1	0	0	0	9	6480	5287	6686	6698
n25m15Q10	26	15	10	-	-	-	-	-	-	-	-	-	-	-	5	0	0	0	1	115	153	tl	tl
n25m15Q15	26	15	15	7.7	15.3	20.3	17.5	10.5	16.7	8.2	**6.0**	13.2	9.0	0	5	0	0	0	2	4071	4540	tl	tl
n25m15Q20	26	15	20	9.6	15.2	16.8	13.2	7.6	13.1	6.8	**4.5**	11.5	7.1	0	3	0	0	0	7	6340	5781	tl	tl
n25m15Q25	26	15	25	10.5	14.1	14.5	10.9	5.4	10.7	6.1	**3.0**	10.6	6.2	0	2	0	0	0	11	6665	5853	tl	tl
n25m15Q30	26	15	30	10.6	13.7	13.8	10.0	4.6	9.9	6.4	**3.0**	11.2	9.2	0	1	0	0	0	19	6802	6102	tl	tl
m5Q5	12	5	5	0.7	1.4	2.2	1.6	1.4	1.0	0.6	0.5	**0.0**	**0.0**	0	0	0	0	0	0	0	0	0	0
m5Q10	12	5	10	2.1	2.4	3.0	2.3	0.8	1.4	0.1	0.1	**0.0**	**0.0**	0	0	0	0	0	0	0	0	0	0
m5Q15	12	5	15	1.6	1.6	1.6	0.9	0.6	0.8	0.1	0.1	**0.0**	**0.0**	0	0	0	0	0	0	0	0	1	0
m10Q5	22	10	5	4.6	2.4	10.4	9.7	7.3	7.3	4.5	3.4	0.9	**0.4**	0	0	0	0	0	0	0	0	1941	2262
m10Q10	22	10	10	8.8	9.9	17.9	16.7	12.7	15.3	10.6	8.5	9.6	**7.7**	0	2	0	0	0	0	26	28	tl	tl
m10Q15	22	10	15	8.6	10.5	12.3	10.4	8.8	10.1	6.8	**6.2**	7.1	6.5	0	1	0	0	0	1	359	398	5802	5796
m10Q20	22	10	20	6.2	8.7	8.9	6.8	5.4	6.6	3.5	**2.9**	4.3	3.8	0	1	0	0	0	2	2290	2865	5863	5861
m10Q25	22	10	25	5.7	8.4	8.5	6.4	4.8	6.1	3.1	**2.6**	4.0	3.6	0	0	0	0	0	4	5450	5258	5942	5927
m10Q30	22	10	30	5.8	8.2	8.2	6.1	4.5	5.9	2.9	**2.4**	3.8	3.4	0	0	0	0	0	5	5460	5254	6063	6357
m15Q5	32	15	5	7.4	**2.9**	14.8	14.4	10.5	11.9	8.6	6.5	8.5	6.5	1	6	0	0	0	0	1	1	tl	tl
m15Q10	32	15	10	9.2	10.5	20.6	18.7	10.8	16.7	12.8	**8.1**	15.1	9.7	0	23	0	0	1	1	76	81	tl	tl
m15Q15	32	15	15	9.2	11.8	15.4	13.4	10.2	12.8	9.7	**8.2**	11.9	10.8	0	13	0	0	0	3	1255	1414	tl	tl
m15Q20	32	15	20	10.4	13.2	14.7	12.9	10.5	12.4	9.2	**8.3**	18.6	22.1	0	7	0	0	0	8	6903	tl	tl	tl
m15Q25	32	15	25	9.1	11.7	12.1	10.2	8.3	9.8	7.2	**6.6**	68.0	62.8	0	4	0	0	0	15	6843	6536	tl	tl
m15Q30	32	15	30	8.9	11.9	11.9	10.0	8.2	9.6	7.9	**7.3**	92.7	100.0	0	4	0	0	0	21	6690	6484	tl	tl

approach (BE) based on the MCF model by Hernández-Pérez & Salazar-González (2009) (HS) also contains heuristic elements. In the cutting plane algorithm for computing the LP relaxation we set $\Delta_G = \Delta_{G_\mathrm{L}} = \Delta_{G_\mathrm{PL}} = 1$ 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 $(c_\mathrm{OPT} - c_\mathrm{LP})/c_\mathrm{OPT}$. 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 loose 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 compensated for most of the instances, except for very tight vehicle capacities. It can be clearly seen that the layered graph models obtain significantly lower 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 $\Delta_G = 0.75, \Delta_{G_\mathrm{L}} = \Delta_{G_\mathrm{PL}} = 0.25$. Let c_LB and c_UB be the best global lower and upper bounds, respectively, obtained by the algorithm within the time limit. The gaps in the tables are given by $(c_\mathrm{UB} - c_\mathrm{LB})/c_\mathrm{UB}$. 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.

For class 2 and 3 instances with a large number of commodities the de-

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.

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

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.

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

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

				LB		UB		time in seconds				
Instance	$\|V\|$	$\|A\|$	$\|\tilde{R}\|$	**BK**	**BCx**	**BK**	**BCx**	**BK**	**BC1**	**BC2**	**BC3**	**BC4**
ESC07	9	40	6	2125	2125	2125	2125	0	0	0	0	0
ESC12	14	132	7	1675	1675	1675	1675	0	0	0	0	0
ESC25	27	622	9	1681	1681	1681	1681	1	**0**	**0**	**0**	**0**
ESC47	49	2187	10	1288	1288	1288	1288	28	7	8	4	**2**
ESC63	65	3613	95	62	62	62	62	**0**	9	2	1	1
ESC78	80	5550	77	18230	18230	18230	18230	**1**	770	46	8664	7569
br17.10	18	237	10	55	55	55	55	**0**	1	**0**	1	**0**
br17.12	18	223	12	55	55	55	55	**0**	1	**0**	1	**0**
ft53.1	54	2722	12	7531	7531	7531	7531	6768	183	218	**91**	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
ft70.1	71	4783	17	39313	39313	39313	39313	363	27	48	**17**	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**
kro124p.1	101	9814	25	37861	**38762**	39420	39420	-	-	-	-	-
kro124p.2	101	9738	49	38809	**39841**	41336	41336	-	-	-	-	-
kro124p.3	101	9339	97	41578	**43904**	**49499**	49570	-	-	-	-	-
kro124p.4	101	5260	131	65445	**73021**	76103	76103	-	-	-	-	-
p43.1	44	1778	9	28140	28140	28140	28140	288	4	**2**	6	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
prob.42	42	1596	10	243	243	243	243	145	**6**	15	8	9
prob.100	100	9579	41	1027	**1045**	**1163**	1346	-	-	-	-	-
rbg048a	50	1569	192	351	351	351	351	21	1	1	1	**0**
rbg050c	52	1703	256	467	467	467	467	3	**1**	**1**	**1**	**1**
rbg109a	111	1748	622	1038	1038	1038	1038	13979	2	**1**	2	2
rbg150a	152	2647	952	1748	**1750**	1750	1750	-	4	**2**	**2**	**2**
rbg174a	176	3309	1113	2033	2033	2033	2033	632	9	11	**6**	**6**
rbg253a	255	5125	1721	2940	**2950**	2950	2950	-	122	107	22	**16**
rbg323a	325	10021	2412	3137	**3140**	3140	3140	-	1714	745	**159**	193
rbg341a	343	9884	2542	2543	**2568**	2568	2568	-	-	-	**70997**	-
rbg358a	360	17998	3239	2529	**2545**	2545	2545	-	20127	12504	2179	**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

mand 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.

8.2. Results for the TSPPC

As mentioned before, relaxing the capacity constraints in the m-PDTSP leads to the TSPPC which is equivalent to the sequential ordering problem. Therefore, we also provide branch-and-cut results on benchmark instances for the TSPPC. We removed all parts from model CUTR* which are only relevant in the capacitated case, i.e., the flow system on the f-variables. The CPU time limit is extended to 1 day.

Table 6 shows branch-and-cut results for instances for the TSPPC 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 & Hoeve, 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 (39) 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. Since the initial model is quite small and the cutting plane only adds violated inequalities even large instances with hundreds of nodes could be solved to optimality ("rbg"-instances). Inequalities (39) used in BC1-2 are able to close the gap for instances with a large number of 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 also considering capacities and thus leading to unnecessary long CPU times in some cases. However, heuristics for the TSPPC 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 (39) 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.

9. 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 and have taken advantage of this relation to provide models for the m-PDTSP that are built by combining two

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.)

			LB			**UB**			**time in seconds**		
Instance	$\lvert A\rvert$	$\lvert\tilde{R}\rvert$	**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**	*29685*	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**

different blocks: 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.

Dumitrescu, I., Ropke, S., Cordeau, J.-F., & Laporte, G. (2010). The traveling salesman problem with pickup and delivery: polyhedral results and a branch-and-cut algorithm. *Mathematical programming*, *121*, 269–305.

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*, *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). 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.

Letchford, A. N., & Salazar-González, J.-J. (2005). 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.

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.

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. J. 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.