The multi-commodity one-to-one Pickup-and-Delivery Traveling Salesman Problem

Hipólito Hernández-Pérez and Juan-José Salazar-González

DEIOC, Facultad de Matemáticas, Universidad de La Laguna,
38271 La Laguna, Tenerife, Spain,
e-mail: {hhperez,jjsalaza}@ull.es

Abstract

This paper treats of a generalization of the Traveling Salesman Problem (TSP) called Multi-commodity one-to-one Pickup-and-Delivery Traveling Salesman Problem (m-PDTSP) in which cities corresponds to customers providing or requiring known amounts of m different objects, and the vehicle has a given upper-limit capacity. Each object has exactly one origin and one destination, and the vehicle must visit each customer exactly once. This justifies the words "one-to-one" and "traveling salesman problem" in the name of the problem, respectively. We introduce a Mixer Integer Linear Programming model for the m-PDTSP, discuss decomposition techniques and describe some strategies to solve the problem based on a branch-and-cut procedure. Preliminary computational experiments on randomly generated euclidian instances are shown.

Key words: Traveling Salesman; Pickup-and-Delivery; Branch-and-Cut; Dial-a-Ride.

1 Introduction

Many practical applications in transportation involve routing and delivery optimization problems. This paper considers the following particular problem. A collection of objects is located at points (sources) in a space, and each one is associated to another point (destination) where it must be delivered. A vehicle is available in a specific point (depot) to carry out the collections and deliveries of the objects. Each object is associated to a weight and the vehicle is associated with a weight capacity. The travel distance between two points is also known. Then the *multi-commodity one-to-one pickup-and-delivery traveling salesman problem* (m-PDTSP) is the problem of finding a route for the vehicle starting and ending at the depot such that it pickups and deliveries all the objects satisfying the capacity limitation and minimizing the total travel distance.

We will assume in this paper that once an object is loaded on the vehicle then it stays on it until it is delivered to its destination. In other words, an object cannot be dropped at intermediate locations and picked up again later by the vehicle. This is named the *non-preemptive* version of the m-PDTSP.

Another hypothesis in this paper is that a destination point of an object is allowed to be the source point of another object, but the same location point is not allowed to be visited more than once by the vehicle. This implies that the route of the vehicle must be a simple cycle and justifies the words *traveling salesman* in the name of the problem, thus means that it is a single-vehicle problem.

Our work approaches the m-PDTSP where the object weights and the vehicle capacities are general numbers. We keep the assumption that all the objects are different, thus the term *multi-commodity* in the name of the problem. See Chalasani and Motwani [12], Anily and Bramel [1] and Hernández and Salazar [22,23] for articles on the case where all the objects are identical (i.e., only one commodity is moved by the vehicle from many sources to many destinations). We also keep the assumption that each object is associated with exactly one source and exactly one destination, thus the term *one-to-one* in the name of the problem. Still, the mathematical model and the algorithm can be easily extended to address the more general problem for the "multi-commodity many-to-many" version of this TSP with pickups and deliveries.

The literature contains many excellent articles on similar problems involving the delivery of different commodities (objects or persons) with a single vehicle. The problems are typically termed *Dial-a-Ride problems* when they also consider time-window constraints (see, e.g., Psaraftis [30,32]). There are a lot variants depending on different requirements, features and optimization functions. We next describe some variants of the DARP.

In the most simplest capacitated version, the *Capacitated Dial-a-Ride Problem* (CDARP) where the a single vehicle should move one unit of a commodity (say, a person) from its origin to its destination with a limited capacity Q and time windows are not considered. Psaraftis [30] gives an dynamic programming algorithm. He also considers a the version where the requests are introduced in the model in real-time (dynamic version of the DARP). Guan [20] studies the preemption and non-preemption CDARP on some graphs (paths, cycles and trees).

When the capacity $Q = 1$, the non-preemptive CDARP is known as the *Stacker Crane Problem*. Fredericson, Hecht and Kim [18] show a worst-case heuristic algorithm for this problem.

When there is no vehicle capacity, the non-preemptive CDARP is called the *Pickup and Delivery Traveling Salesman Problem* (PDTSP). The PDTSP assumes that one vertex is only a origin or a destination of a commodity. Some references about PDTSP are Kalantari, Hill and Arora [26], Kubo and Kasugai [27], Healy and Moll [21], Renaud, Boctor and Ouenniche [34], Renaud, Boctor and Laporte [33], Ruland [37] and Ruland and Rodin [38]. In [26] the PDTSP is referred to as TSPPD, in [21] it is referred to as CDARP and in [37] and [38] it is referred as *Pickup and Delivery Problem* (PDP). The problem which one vertex can be the origin and/or destination of various commodities is called *TSP with Precedence Constraints* (TSPPC). Bianco, Mingozzi, Riccardelli and Spadoni [6] describes a dinimic programming algorithm for this problem. The asymmetric version, called the *Asymmetric TSP with Precedence Constrains* (ATSPPC) or *Sequential Ordering Problem*, have been studied by Balas, Fischetti and Pulleyblank [5], Ascheuer, Jünger and Reinelt [4] and Gouveia and Pesneau [19]. The m-PDTSP is the capacitated version of the ATSPPC.

Most of articles on the Dial-a-Ride problem consider additional features as multi-vehicle version, time windows and/or dynamic requests. Moreover, the objective function is not always to minimize the total cost of the routes, but the *inconvenience* of the users with a function of the waiting and ride times. Also, minimizing the number of vehicles is an objective in some versions of the multi-vehicle DARP.

All the following references for the DARP consider some kind of time windows. In our knowledge, the unique DARP with time windows that consider dynamic requests is shown in Madsen, Ravn and Rygaard [28] where a heuristic algorithm for a multi-vehicle version is described. When the request are known beforehand there are several references. Psaraftis [32,31] studies the problem with time windows and he shows an exact and a heuristic algorithms for a single vehicle. Sexton and Bodin [39,40] give heuristic procedures and Desrosiers, Dumas and Soumis [17] present an exact algorithm based on dynamic programming for the single vehicle version also. Heuristic algorithms for the multi-vehicle version are presented in Jaw, Odoni, Psaraftis and Wilson [25], Bodin and Sexton [8], Ioachim, Desrosiers, Dumas and Solomon [24], Toth and Vigo [42,43], Borndörfer, Grötschel, Klostemeier and Küttner [9], Wolfler-Calvo and Colorni [44], and Cordeau and Laporte [13]. A recent article Cordeau [14] presents a branch-and-cut algorithm for the multi-vehicle DARP with time windows. Cordeau also considers the transportation of group of people (i.e., the the quantity demanded is grater than one). Finally, we reference the survey of Courdeau and Laporte [15] where articles which deal with DARP are classified.

The multi-vehicle version of the DARP is known as *Pickup and Delivery Problem* (PDP). Many times this problem has other features as *time windows* and *precedence constraints*. Recent works about that are Sigurt, Pisinger and Sig [41], Ropke and Pisinger [36] and Ropke, Cordeau and Laporte [35].

Anily and Hassin [3] introduce the *Swapping Problem*, a more general problem where several commodities must be transported from many origins to many destinations with a vehicle of limited capacity following a non-necessary Hamiltonian route, and where a commodity can be temporarily stored in an intermediate customer (i.e., preemption is allowed). Anily and Hassin [3] present an worst-case heuristic algorithm for the Swapping Problem with a ratio of 2.5 based on an algorithm for the TSP. The Swapping Problem on a line is analyzed in Anily, Gendreau and Laporte [2].

This article is structured as follows. Section 2 describes a mathematical models and a decomposition method for the m-PDTSP. Section 3 shows valid inequalities for these models. Section 4 shows some computational results of four algorithms for the m-PDTSP. Finally, Section 5 proposes further extension of this methodology.

2 Mathematical Models

This section presents a Mixed Integer Linear Programming (MILP) formulation for the m-PDTSP and some variations of this model. Let us start by introducing some notation. The depot will be represented by two nodes 0 and $n+1$, and customers by a vertex i, for $i = 1, \ldots, n$. Thus, $V := \{0, 1, \ldots, n, n+1\}$ is the vertex set of a complete directed graph $G = (V, A)$. Note that a customer i can be source of various commodities and the destination of other commodities. For each pair of locations $(i, j) \in A$, the travel distance (or cost) c_{ij} of going from i to j is given. There are m commodities (objects), each one $k \in K := \{1, \ldots, m\}$ associated with a weight q_k, with a source $s_k \in \{0, 1, \ldots, n\}$ and with a destination $d_k \in \{1, \ldots, n, n+1\}$. The capacity of the vehicle is represented by Q and is assumed to be a positive number. For any subset $S \subset V$, let $\delta^+(S) := \{(i, j) \in A : i \in S, j \notin S\}$, $\delta^-(S) := \{(i, j) \in A : i \notin S, j \in S\}$ and when $S = i$ (with only one element) we write $\delta^+(i)$ and $\delta^-(i)$.

The m-PDTSP is the problem of finding a Hamiltonian path (i.e., a path traversing each node exactly once) from 0 to $n+1$ such that all the commodities are collected and delivered without ever violating the vehicle capacity and minimizing the total distance.

For each node $i \in V$, let $K_i^+ := \{k \in K : s_k = i\}$, $K_i^- := \{k \in K : d_k = i\}$, $q_i^+ := \sum\{q_k : k \in K_i^+\}$ and $q_i^- := \sum\{q_k : k \in K_i^-\}$. The source and destination of each commodity imply a precedence constraint between some pairs of nodes in V. Because each node in the graph G must be visited at most once, it is fundamental for the existence of a m-PDTSP solution that the precedence constraints induce an acyclic subgraph in G (see Bianco, Mingozzi, Riccardelli and Spadoni [6]). If each node is the source of at most one commodity and/or the destination of at most one commodity, and if the precedence constraints induce an acyclic subgraph in G then condition

$$Q \geq \max\{q_i^+, q_i^- : i \in V\} \tag{1}$$

is a necessary and sufficient condition for the existence of a m-PDTSP solution. Similarly, if $K_0^+ \cup K_{n+1}^- = K$ then $\max\{q_i^+, q_i^- : i \in V\} = \max\{q_k : k \in K\}$, and therefore the condition (1) is also necessary and sufficient for the m-PDTSP feasibility (for example, the Hamiltonian path that visits the customers in $\{d_k : k \in K_0^+\}$ before any customer in $\{s_k : k \in K_{n+1}^-\}$).

2.1 A Mixed Integer Linear Programming

To provide a mathematical model to m-PDTSP, for each arc $a \in A$, we introduce a 0-1 variable

$$x_a := \begin{cases} 1 & \text{if and only if } a \text{ is routed,} \\ 0 & \text{otherwise,} \end{cases}$$

and the 0-1 variable

$$f_a^k := \text{the load of commodity } k \text{ in the vehicle when routing } a.$$

For simplicity in next notation, we assume that $K_0^+ = K_{n+1}^- = \emptyset$ (i.e., the depot does not give or receive any load). Hence, the flow-variables f_a^k can be fixed to zero for all $k \in K$ and all $a \in \delta^+(0) \cup \delta^-(n+1)$. The m-PDTSP can be formulated as:

$$\min \sum_{a \in A} c_a x_a \tag{2}$$

subject to

$$\sum_{a\in\delta^-(\{i\})} x_a = 1 \quad \text{for all } i \in V\setminus\{0\} \tag{3}$$
$$\sum_{a\in\delta^+(\{i\})} x_a = 1 \quad \text{for all } i \in V\setminus\{n+1\} \tag{4}$$
$$\sum_{a\in\delta^-(S)} x_a \geq 1 \quad \text{for all } S\subset V : 0\notin S \tag{5}$$
$$\sum_{a\in\delta^+(S)} x_a \geq 1 \quad \text{for all } S\subset V : n+1\notin S \tag{6}$$
$$x_a \in \{0,1\} \quad \text{for all } a\in A \tag{7}$$
$$\sum_{a\in\delta^+(\{i\})} f_a^k - \sum_{a\in\delta^-(\{i\})} f_a^k = \begin{cases} q_k & \text{if } i = s_k \\ -q_k & \text{if } i = d_k \\ 0 & \text{otherwise} \end{cases} \quad \text{for all } i\in V \text{ and } k\in K \tag{8}$$
$$0 \leq \sum_{k\in K} f_a^k \leq Q x_a \quad \text{for all } a\in A. \tag{9}$$

Constraints (3)–(7) impose that x must represent a simple cycle from 0 to $n+1$. Such a path is a m-PDTSP solution when there exists a path from s_k to d_k for each commodity k as represented by equations (8) such that the capacity constraint (9) is satisfied. It is not necessary to impose the integrality conditions on the variables f_a^k defining the path for each commodity.

This mathematical model is a MILP model and it has a large number of variables due to the f_a^k variables. Therefore, it is unlikely that a general-purpose MILP solver (like `Cplex`) can use it to solve the m-PDTSP with a medium number of commodities and location points. However, we are going to show another variants to try to solve the m-PDTSP.

2.2 Fixing Variables and Strengthen the MILP Model

Of course, some variables can be fixed to zero. Taking account the precedence constraints that induce this problem, an arc variable x_{ij} can be fixed to zero if:

- $i=0$, $s_k \neq 0$ and $j = d_k$,
- $i = s_k$, $j = n+1$ and $n+1 \neq s_k$, or
- $i = d_k$ and $j = s_k$.

Looking a the capacity of the vehicle, a arc variable $x_{i,j}$ can be fixed to zero if

- $\sum_{i=d_k \vee (j=d_k \wedge i \neq s_k)} q^k > Q$,
- $\sum_{i=s_k \vee j=d_k} q^k > Q$, or
- $\sum_{(i=s_k \wedge j \neq d_k) \vee j=s_k)} q^k > Q$.

This quantities are lower bounds of the load of the vehicles going into, throwing and going out (i, j) if the arc (i,j)is routed, respectively.

Obviously, if $x_{ij} = 0$ then $f_{ij}^k = 0$ for all k. But f_{ij}^k can be fixed to zero if:

- $i = 0$ and $i \neq s_k$,
- $j = n + 1$ and $j \neq d_k$,
- $i = d_k$, or
- $j = s_k$.

We present a strengthen the of the relaxation of model (3)–(9). That is, we can see that some flow-variables can be "fixed":

$$f_{ij}^k = q^k x_{ij} \qquad \text{if } i = s_k \text{ or } j = d_k. \tag{10}$$

Similarly to other capacitated transportation problems (as in the Capacitated Vehicle Routing Problem) when they are formulated with flow (or load) variables, the capacity constraint (9) can be strengthen. That is, given an arc $a = (i, j)$ the load the vehicle cannot be greater than $Q - \sum_{d_k=i} q_k + \sum_{s_k=i}$ (the vehicle have to have enough free space to pickup the load of customer i) and cannot be greater than $Q - \sum_{s_k=j} q_k + \sum_{d_k=j} q_k$ (the vehicle have to have enough free space to pickup the load of customer j). Thus inequality

$$\sum \sum_{k \in K} f_{ij}^k \leq x_k \left(Q - \max \left\{ 0, \sum_{d_k=i} q_k - \sum_{s_k=i} q_k, \sum_{s_k=j} q_k - \sum_{d_k=j} q_k \right\} \right) \quad \text{for all } (i, j) \in A. \tag{11}$$

is valid.

2.3 *Projecting out the Flow Variables*

From the model (3)–(9), we can the flow variables (i.e., equations (8) and (9)) if we insert appropriate constraints on the x variables. That is, if we have an algorithm to find an violated inequalities on the x ...

2.4 A "Double" Decomposition Technique

This section shows a procedure to find an optimal solution of the model (2)–(9) without explicitly working with a single program with all the variables. The idea uses the fact that the f_a^k variables are continuous variables whose existence certificates that a route described by x is feasible for the m-PDTSP. Then, by using Farkas' Lemma in Linear Programming, it is possible to work with only the variables x_a. The procedure follows the classical scheme of the Benders' Decomposition (Benders [10]) and Dantzig-Wolfe Decomposition (Dantzig and Wolfe [16]), and it is next described.

Let us consider a vector x' satisfying (3)–(6). In order to have a guarantee that it is an m-PDTSP solution, constraints (8)–(9) require the existence of the variables f_a^k defining the paths to move each commodity on the route. An alternative way of writing this requirement on x' is the following. Let $\mathcal{P}^k$ be the collection of all paths in G from s_k to d_k. For each path $p \in \mathcal{P}^k$ let $z^p = [z_a^p : a \in A]$ be a 0-1 vector such that

$$z_a^p := \begin{cases} 1 & \text{if and only if arc } a \text{ is in path } p, \\ 0 & \text{otherwise.} \end{cases}$$

Then, x' is a feasible solution for the m-PDTSP if and only if there is a solution λ for the following linear system:

$$\sum_{k \in K} q_k \sum_{p \in \mathcal{P}^k : z_a^p = 1} \lambda_p \le Q x'_a \quad \text{for all } a \in A, \tag{12}$$

$$\sum_{p \in \mathcal{P}^k} \lambda_p = 1 \quad \text{for all } k \in K, \tag{13}$$

$$\lambda_p \ge 0 \quad \text{for all } p \in \mathcal{P}^k \text{ and all } k \in K. \tag{14}$$

This system impose that there must be a path for each commodity such that all together follow the route defined by x' satisfying the capacity constraint on each arc.

Clearly, the system (12)–(14) has much more variables than the system (8)–

(9), which is a disadvantage for checking the m-PDTSP feasibility of a given vector $x = x'$. Nevertheless, by using Farkas Lemma the vector x' will be m-PDTSP feasible if and only if all the rays $(u, v) = [u_a : a \in A; v_k : k \in K]$ in the cone

$$q_k\Big(\sum_{a\in A: z_a^p=1} u_a\Big) + v_k \geq 0 \quad \text{for all } k \in K, p \in \mathcal{P}^k, \tag{15}$$

$$u_a \geq 0 \quad \text{for all } a \in A, \tag{16}$$

$$v_k \leq 0 \quad \text{for all } k \in K, \tag{17}$$

satisfy

$$\sum_{a\in A} Qx'_a u_a + \sum_{k\in K} v_k \geq 0.$$

The variable u_a represents the dual variable associated to each inequality in (12) and v_k represents the dual variable associated to each equation in (13). Note that we can restrict v_k to be non-positive since each equation in (13) can be replaced by the inequality $\sum_{p\in\mathcal{P}^k} \lambda_p \geq 1$ without loss of generality.

Now the feasibility check is transformed into the optimization problem of minimizing $\sum_{a\in A} Qx'_a u_a + \sum_{k\in K} v_k$ over the constraints (15)–(17). At first glance one cannot see any advantage on this new reformulation due to the large number of constraints in (15). Fortunately, it is not a disadvantage since the resolution of this problem can be replaced by an iterative procedure where a relaxed problem (i.e., a problem with only some constraints) is solved and possibly strengthened with some missing constraints. More precisely, at each iteration, the problem is determined by a subset of paths $\mathcal{Q}^k \subseteq P^k$ for each commodity $k \in K$. If it is not unbounded then 0 is the optimal solution of the full problem, and therefore the vector x' is m-PDTSP feasible. Otherwise, there is a ray (u', v') such that $\sum_{a\in A} Qx'_a u'_a + \sum_{k\in K} v'_k < 0$ and we need to check if a path in $\mathcal{P}^k \setminus \mathcal{Q}^k$ is necessary. This check consists on finding (if any) a path $p \in \mathcal{P}^k$ such that

$$\sum_{a\in A: z_a^p=1} u'_a \leq -v'_k / q_k$$

which can be done by solving a *Shortest Path Problem* from s_k to d_k in G where the distance of an arc a is given by the u'_a. Let p' be a shortest path and l' the total distance of p'. If $q_k l' + v'_k < 0$ then we must enlarge $Q^k := Q^k \cup \{p'\}$

and consider a new iteration with an strengthened problem. Otherwise, no path is missing from the current problem and therefore

$$\sum_{a \in A} Q u'_a x_a \geq - \sum_{k \in K} v'_k \tag{18}$$

is a valid inequality violated by x' that must be considered in addition to (3)–(6). The inequality (18) is termed *Benders' cut* since it is derived as in the classical Benders' decomposition approach to a MILP model. The problem of checking if there exists a violated Benders' cut for a given x' is termed *separation problem*, and its resolution requires a column-generation algorithm if the problem is formulated using (12)–(14), or a cutting-plane approach if the problem is formulated using (15)–(17).

The above scheme leads to a decomposition technique for solving the mathematical model (2)–(9). Instead of solving the full model with all the f_a^k variables, the technique consists of solving a 0-1 integer linear programming model with the binary variables x_a, minimizing (2) subject to constraints (3)–(6) and (18) for all (u', v') rays for the cone (15)–(17). To solve this problem in a more efficient way, we propose a branch-and-cut procedure (see, e.g., Caprara and Fischetti [11]) where two optimization problems are iteratively solved. The first problem is named *master problem* and it is the linear relaxation of the problem minimizing (2) subject to (3)–(6) and some (18). The second problem is named *subproblem* and it is used to generate missed constraints (18) as they are necessary. At each iteration the master problem produces an optimal (likely non-integer) solution x', and the subproblem finds (if any) a violated Benders' cut inequality. When no violated Benders' cut is generated by the subproblem, then the whole procedure stops with the optimal solution x' if it is an integer vector, or the procedure requests a branching phase to continue (possibly) generating new Benders' cuts.

A non-standard idea in this Benders' decomposition approach is that the subproblem can be applied when x' is a non-integer vector. In this way, each master problem is a linear program and therefore it can be solved with less computational effort. The counterpart is that a branching procedure is required to further guarantee the integrality of the solution. Alternatively, another approach based on the same decomposition could solve the master problem with the integrality constraints on the variables, and then the whole procedure would be a cutting-plane approach. Clearly, this second approach consumes more computational effort to solve the master problem at each iteration. We have conducted experiments with both approaches and they are discussed in the next section.

3 Valid inequalities

As the m-PDTSP is a capacitated version of the ATSPPC, all valid inequalities for the ATSPPC are valid for the m-PDTSP. We show some of them implemented in the algorithms. Also we show inequalities derived from the capacity requirement.

...

3.1 Capacity Constraints

$$x(\delta^{+}(S)) \geq r(S) \qquad \text{for all } S \subset V \text{ and } 0, n+1 \in V \setminus S, \tag{19}$$

where

$$r(S) = \max\{1, \frac{1}{Q} \sum_{s_k \in S,\ \mathrm{d}_k \notin S} q_k, \frac{1}{Q} \sum_{s_k \notin S,\ \mathrm{d}_k \in S} q_k\}$$

is a lower bound of the number of times that the vehicle has to go inside/outside the customer set S. Observe that $\sum_{s_k \in S,\ \mathrm{d}_k \notin S} q_k$ is a lower bound for the load transported from S to $V \setminus S$ and $\sum_{s_k \notin S,\ \mathrm{d}_k \in S} q_k$ is a lower bound for the loas transported from $V \setminus S$ to S.

3.2 Predecessor and Successor Inequalities

For each commodity k there is a precedent relation between s_k and d_k. In other words, s_k have to visited before d_k. This condition can used to strengthen the subtour elimination constraints (see, e.g., Balas, Fischetti and Pulleyblank [5]). Hence teh following inequalities are valid for the one-to-one m-PDTSP:

!!!! Ojo hay que unificar la notación !!!!

The predecessor-inequalities

$$x(S \setminus \pi(S), \overline{S} \setminus \pi(S)) \quad \geq 1 \quad \text{for all } S \subset V \setminus \{0, n+1\}, \tag{20}$$

where $\pi(S)$ denotes the set of predecessor customers of S and $\overline{S}$ the customers not in S (including 0 and $n+1$).

The above inequalities can be rewritten as

$$x(S \setminus \pi(S), S \cup \pi(S)) \quad \leq |S \setminus \pi(S)| \quad \text{for all } S \subset V \setminus \{0, n+1\},$$

because of the degree equations (3) and (4).

The successor-inequalities

$$x(\overline{S} \setminus \sigma(S), S \setminus \sigma(S)) \quad \geq 1 \quad \text{for all } S \subset V \setminus \{0, n+1\}, \tag{21}$$

where $\sigma(S)$ denotes the set of successor customers of S and $\overline{S}$ the customers not in S (including 0 and $n+1$).

3.3 The Precedence Cycle Breaking Inequalities

Let $S_1, \ldots, \S_l \subset V \setminus \{0, n+1\}$ ($l \geq 2$) disjoint node sets such that

$$\sigma(S_i) \cap S_{i+1} \neq \emptyset$$

for $i = 1, \ldots, l$ (or equivalently $S_i \cap \pi(S_{i+1}) \neq \emptyset$ for $i = 1, \ldots, l$) with $S_{l+1} = S_1$. Then the inequality

$$\sum_{i=1}^{l} x(S_i, S_i) \leq \sum_{i=1}^{l} |S_i| - l - 1 \tag{22}$$

is valid for the m-PDTSP. See, for example [5] for details and a proof for the ATSPPC.

A simple form of (22) arises when $l = 2$ and $|S_2| = 1$, in this case the inequality becomes

$$x(S_1, S_1) \leq |S_1| - 2$$

. This is a strengthen of the subtour elimination constraints for S_1.

4 Computational Results

The algorithms described in the previous section have been implemented in a computer program and tested on solving some randomly-generated instances. This section presents the results of the conducted experiments.

The algorithms were written using C programming language in a Personal Computer with AMD Athlon 2.0 Ghz and running `Windows XP`. We used the `Cplex 9.0` branch-and-cut framework for embedding our code implementations.

Since there are no benchmark instances in the literature for the m-PDTSP, we have transformed the proposed for the ATSPPC (*Class 1*) in he following way. Each precedence relation (i, j) generates a commodity (say) k. Thus commodity k must be moved from i to j. There are considered two different transformations depending on the values of q_k. Fist q_k is fixed to 1 for all commodities and second q_k is a random number in $[1, 5]$. Finally, different capacities of the vehicle Q are considered also. Thus, each instance for the ATSPPC correspond to many instances for m-PDTSP with different q_k (fixed and random) and different Q.

Also, we have using the following generator (*Class 2*), similar to the described in Mosheiov [29] for the TSP with Pickups and Deliveries. We have generated n random points in the square $[-500, 500] \times [-500, 500]$ which correspond to the customers. The depot is located at point (0,0). The travel cost c_{ij} between points i and j was computed as the Euclidean distance between the two location points. The relations of origins and destination are generated as in [4]. Iteratively, the generator chooses an arc (i, j); if it is not generates a cycle, the process is repeated until m relation are generates. The number of commodities m is in $\{5, 10, 15\}$. Also different capacity Q for each m are considered ($Q \in \{500, 50, 45, \ldots, 15\}$).

Finally, a third class is generated (*Class 3*). The vertices are generated as the

Class 2 but each vertex is the origin or destination of a unique commodity. Thus, vertex i is the depot if $i = 0$ or $i = n + 1$, i is the origin of commodity k if $i = 2 * k + 1$ and i is the destination of commodity k if $i = 2 * k + 2$. The quantities q_k are randomly generated integers in $[1, 5]$. Different capacities Q are considered also.

4.1 Testing different strategics

Each instance was solved with these approaches:

Alg. 1: Model (3)–(9) and (10).
Alg. 2: Model (3)–(9), (10) and (11).
Alg. 3: The projected model and where the subproblem is solved as the decomposition technique described in Subsection .
Alg. 4: The projected model and where detected cuts are inserted.

Table 1 shows the average computational results of the different strategies. Each row corresponds to the results of ten instances. The first three columns correspond to the number of commodities m, the number of customers including the depot n, and the vehicle capacity Q. The following columns correspond to the average computational time (*time*), the number of instances that the algorithm does not find an integer solution before the time limit (*inf.*) and the number of instances that do not end before the time limit but the algorithm finds an integer solution (*t.l.*) for each strategy. The average computational time is computed only for the instances that finish before the time limit and (3600 seconds).

Unfortunate, the above described decomposition technique is not good solving the one-to-one m-PDTSP. The classical Benders decomposition gains the other algorithms solving the one-to-one m-PDTSP.

!!! Se podrían poner tb las LB al nodo raíz peor son muy similares por los tres algoritmos¡¡¡

Table 1
Computational results of Class 1

name	m	n	Q	Alg. 1 time	Alg. 1 inf.	Alg. 1 t.l.	Alg. 2 time	Alg. 2 inf.	Alg. 2 t.l.	Alg. 3 time	Alg. 3 inf.	Alg. 3 t.l.	Alg. 4 time	Alg. 4 inf.	Alg. 4 t.l.
ESC07Q3max1	6	9	3	0.22	0	0	0.23	0	0	0.14	0	0	0.14	0	0
ESC07Q15max5	6	9	15	0.20	0	0	0.23	0	0	0.13	0	0	0.13	0	0
ESC11Q3max1	3	13	3	0.20	0	0	0.25	0	0	0.20	0	0	0.22	0	0
ESC11Q15max5	3	13	15	0.22	0	0	0.25	0	0	0.22	0	0	0.19	0	0
ESC12Q6max1	7	14	6	0.41	0	0	0.34	0	0	0.25	0	0	0.22	0	0
ESC12Q5max1	7	14	5	0.80	0	0	1.33	0	0	1.09	0	0	0.36	0	0
ESC12Q4max1	7	14	4	2.63	0	0	1.91	0	0	1.34	0	0	0.42	0	0
ESC12Q20max5	7	14	20	0.48	0	0	0.31	0	0	0.27	0	0	0.22	0	0
ESC12Q15max5	7	14	15	1.61	0	0	1.72	0	0	1.31	0	0	0.34	0	0
br17.10Q5max1	10	18	5	4.19	0	0	1.78	0	0	1.00	0	0	0.06	0	0
br17.10Q4max1	10	18	4	0.00	0	1	0.00	0	1	0.00	0	1	0.00	0	1
br17.10Q3max1	10	18	3	260.31	0	0	483.44	0	0	516.28	0	0	166.59	0	0
br17.10Q15max5	10	18	15	3.19	0	0	1.30	0	0	3.13	0	0	0.30	0	0
br17.12Q5max1	12	18	5	5.14	0	0	11.97	0	0	0.66	0	0	0.01	0	0
br17.12Q4max1	12	18	4	0.00	0	1	0.00	0	1	0.00	0	1	0.00	0	1
br17.12Q3max1	12	18	3	1284.86	0	0	0.00	0	1	0.00	0	1	0.00	0	1
br17.12Q15max5	12	18	15	299.11	0	0	595.47	0	0	66.56	0	0	2.81	0	0
ESC25Q5max1	9	27	5	19.52	0	0	14.44	0	0	470.61	0	0	0.19	0	0
ESC25Q4max1	9	27	4	608.16	0	0	592.31	0	0	0.00	1	0	4.23	0	0
ESC25Q3max1	9	27	3	0.00	0	1	1143.80	0	0	0.00	1	0	85.14	0	0
ESC25Q15max5	9	27	15	16.13	0	0	12.16	0	0	0.00	0	1	0.16	0	0

4.2 Extended computational results for the best strategics

The following three tables (Tables 4.2, 4.2 and 4.2) shows several computational results results of the Algorithm 4 for the three classes of instances.

Table 2
Computational results of Class 2

			Alg. 1			Alg. 2			Alg. 3			Alg. 4		
m	n	Q	time	inf.	t.l.	time	inf.	t.l.	time	inf.	t.l.	time	inf.	t.l.
5	11	500	0.09	0	0	0.11	0	0	0.06	0	0	0.06	0	0
5	11	30	0.09	0	0	0.11	0	0	0.06	0	0	0.05	0	0
5	11	25	0.10	0	0	0.11	0	0	0.06	0	0	0.05	0	0
5	11	20	0.10	0	0	0.11	0	0	0.06	0	0	0.05	0	0
5	11	15	0.09	0	0	0.10	0	0	0.06	0	0	0.05	0	0
5	11	10	0.13	0	0	0.15	0	0	0.07	0	0	0.05	0	0
10	11	500	0.07	0	0	0.07	0	0	0.05	0	0	0.04	0	0
10	11	30	0.06	0	0	0.07	0	0	0.05	0	0	0.04	0	0
10	11	25	0.06	0	0	0.07	0	0	0.04	0	0	0.04	0	0
10	11	20	0.06	0	0	0.07	0	0	0.04	0	0	0.04	0	0
10	11	15	0.08	1	0	0.10	1	0	0.06	1	0	0.05	1	0
10	11	10	0.05	7	0	0.06	7	0	0.04	7	0	0.04	7	0
15	11	500	0.05	0	0	0.04	0	0	0.04	0	0	0.03	0	0
15	11	30	0.03	2	0	0.04	2	0	0.03	2	0	0.03	2	0
15	11	25	0.04	4	0	0.03	4	0	0.03	4	0	0.03	4	0
15	11	20	0.03	6	0	0.04	6	0	0.03	6	0	0.03	6	0
15	11	15	0.02	9	0	0.03	9	0	0.03	9	0	0.03	9	0
15	11	10	0.00	10	0	0.00	10	0	0.00	10	0	0.00	10	0
5	16	500	5.94	0	0	6.44	0	0	67.68	0	0	1.28	0	0
5	16	30	6.48	0	0	7.24	0	0	79.91	0	0	1.27	0	0
5	16	25	5.61	0	0	6.57	0	0	89.83	0	0	1.28	0	0
5	16	20	5.43	0	0	7.28	0	0	128.47	0	0	1.28	0	0
5	16	15	8.05	0	0	11.05	0	0	161.30	0	0	1.28	0	0
5	16	10	4.23	0	0	10.25	0	0	80.79	0	0	1.25	0	0
10	16	500	4.36	0	0	4.49	0	0	7.26	0	0	0.33	0	0
10	16	30	4.49	0	0	5.94	0	0	6.96	0	0	0.32	0	0
10	16	25	4.70	0	0	7.04	0	0	8.43	0	0	0.36	0	0
10	16	20	10.39	0	0	8.77	0	0	20.68	0	0	0.53	0	0
10	16	15	13.34	1	0	17.43	1	0	12.82	1	0	0.62	1	0
10	16	10	18.30	7	0	61.54	7	0	97.72	7	0	2.84	7	0
15	16	500	0.83	0	0	0.87	0	0	0.48	0	0	0.08	0	0
15	16	30	1.05	0	0	1.02	0	0	0.42	0	0	0.08	0	0
15	16	25	1.10	0	0	1.22	0	0	0.46	0	0	0.08	0	0
15	16	20	50.99	2	0	45.41	2	0	410.23	2	0	1.71	2	0
15	16	15	131.26	4	0	261.96	4	0	244.98	4	0	3.90	4	0
15	16	10	0.00	10	0	0.00	10	0	0.00	10	0	0.00	10	0
5	21	500	1.94	0	0	4.20	0	0	91.30	0	0	0.52	0	0
5	21	30	3.77	0	0	3.94	0	0	209.85	0	0	0.40	0	0
5	21	25	6.13	0	0	2.44	0	0	255.34	0	0	0.40	0	0
5	21	20	4.06	0	0	11.10	0	0	46.95	0	0	0.40	0	0
5	21	15	2.64	0	0	2.69	0	0	124.71	0	0	0.40	0	0
5	21	10	30.88	0	0	39.66	0	0	19.37	0	2	10.51	0	0
10	21	500	25.57	0	1	22.67	0	1	277.68	0	1	114.58	0	0
10	21	30	24.61	0	1	33.50	0	1	47.06	0	2	114.69	0	0
10	21	25	28.02	0	1	40.56	0	1	58.88	0	2	114.74	0	0
10	21	20	67.29	0	1	70.28	0	1	506.93	0	2	115.16	0	0
10	21	15	296.89	0	2	460.29	0	2	1026.23	1	3	133.08	0	0
10	21	10	941.55	3	3	720.43	3	2	642.42	4	3	112.08	2	2
15	21	500	927.45	0	2	899.81	0	2	157.99	0	5	204.03	0	0
15	21	30	208.16	0	5	60.06	0	5	770.93	0	5	215.24	0	0

Table 3
Computational results of Class 3

			Alg. 1			Alg. 2			Alg. 3			Alg. 4		
m	n	Q	time	inf.	t.l.	time	inf.	t.l.	time	inf.	t.l.	time	inf.	t.l.
5	12	500	0.34	0	0	0.32	0	0	0.20	0	0	0.20	0	0
5	12	30	0.30	0	0	0.31	0	0	0.21	0	0	0.20	0	0
5	12	25	0.30	0	0	0.32	0	0	0.20	0	0	0.20	0	0
5	12	20	0.30	0	0	0.33	0	0	0.20	0	0	0.19	0	0
5	12	15	0.34	0	0	0.32	0	0	0.21	0	0	0.20	0	0
5	12	10	0.31	0	0	0.37	0	0	0.22	0	0	0.20	0	0
5	12	5	0.14	0	0	0.26	0	0	0.18	0	0	0.16	0	0
10	22	500	147.75	0	0	223.80	0	0	697.44	0	3	3.80	0	0
10	22	30	256.89	0	0	240.84	0	0	258.90	0	6	3.78	0	0
10	22	25	380.05	0	0	241.89	0	0	980.98	0	6	4.50	0	0
10	22	20	579.75	0	0	784.17	0	0	282.66	1	5	10.05	0	0
10	22	15	684.19	0	3	911.46	0	4	702.18	2	5	155.81	0	0
10	22	10	941.87	0	0	1058.80	0	5	1299.95	6	3	478.92	0	0
10	22	5	5.06	0	0	24.54	0	0	3.10	0	1	1.76	0	0

5 Further Extension

The m-PDTSP problem is the relaxed version of the so-called *single-vehicle Dial-a-ride problem* (DARP) where the *time window* and *ride time* requirements are not considered. These requirements are the following. Let t_a be the time for routing the arc $a \in A$, and t_i the time for serving the demand at location point $i \in V$. Let $[e_i, l_i]$ be the time window associated with point i, where e_i and l_i represent the earliest and latest time, respectively. Let r_k the maximum ride time for commodity $k \in K$, i.e. the maximum travel time from when it is collected in $s_k \in V$ to when it is delivered at $d_k \in V$. Then, with these new time constraints, (x', y') is a feasible solution if and only if the vehicle arrives at each node at a time w_i such that $e_i \leq w_i \leq l_i$ and $w_{d_k} - w_{s_k} \geq r_k$ for all $k \in K$.

By adapting the introduced model and algorithm for the m-PDTSP, it is pos-

Table 4
Extended computational results of Class 1 and Alg. 4

				Cuts		Lower bounds				Sep. time					
name	m	n	Q	SEC	others	SEC	others	opt./PC	B&C	SEC	others	root-t	time	unf.	t.l.
ESC07Q3max1	6	9	3	4.0	0.0	100.0	100.0	100.00	0.0	0.00	0.00	0.00	0.13	0	0
ESC07Q15max5	6	9	15	4.0	0.0	100.0	100.0	100.00	0.0	0.00	0.00	0.00	0.13	0	0
ESC11Q3max1	3	13	3	3.0	56.0	97.8	99.2	100.00	2.0	0.00	0.00	0.17	0.20	0	0
ESC11Q15max5	3	13	15	3.0	56.0	97.8	99.2	100.00	2.0	0.00	0.00	0.16	0.19	0	0
ESC12Q6max1	7	14	6	13.0	4.0	98.2	98.5	100.00	2.0	0.00	0.00	0.17	0.20	0	0
ESC12Q5max1	7	14	5	23.0	92.0	84.5	86.2	116.72	145.0	0.02	0.05	0.19	0.34	0	0
ESC12Q4max1	7	14	4	17.0	263.0	78.3	82.2	126.03	137.0	0.03	0.02	0.19	0.42	0	0
ESC12Q3max1	7	14	3	0.0	0.0	0.0	0.0	0.00	0.0	0.00	0.00	0.00	0.00	1	0
ESC12Q20max5	7	14	20	13.0	4.0	98.2	98.5	100.00	2.0	0.00	0.00	0.19	0.22	0	0
ESC12Q15max5	7	14	15	23.0	92.0	84.5	86.2	116.72	145.0	0.00	0.06	0.19	0.34	0	0
br17.10Q5max1	10	18	5	18.0	214.0	96.4	100.0	100.00	3.0	0.00	0.00	0.22	0.25	0	0
br17.10Q4max1	10	18	4	0.0	0.0	0.0	0.0	0.00	0.0	0.00	0.00	0.00	0.00	0	1
br17.10Q3max1	10	18	3	20.0	3298.0	58.5	76.0	149.09	4365.0	0.59	2.78	0.17	166.25	0	0
br17.10Q15max5	10	18	15	21.0	346.0	96.4	100.0	100.00	57.0	0.00	0.09	0.01	0.30	0	0
br17.12Q5max1	12	18	5	26.0	107.0	96.4	100.0	100.00	0.0	0.00	0.00	0.00	0.03	0	0
br17.12Q4max1	12	18	4	0.0	0.0	0.0	0.0	0.00	0.0	0.00	0.00	0.00	0.00	0	1
br17.12Q3max1	12	18	3	0.0	0.0	0.0	0.0	0.00	0.0	0.00	0.00	0.00	0.00	0	1
br17.12Q15max5	12	18	15	43.0	773.0	90.9	90.9	100.00	616.0	0.05	0.41	0.05	2.84	0	0
ESC25Q5max1	9	27	5	8.0	165.0	95.1	99.4	100.00	2.0	0.00	0.00	0.14	0.14	0	0
ESC25Q4max1	9	27	4	30.0	1199.0	83.5	91.0	113.92	262.0	0.06	0.77	0.20	4.22	0	0
ESC25Q3max1	9	27	3	65.0	4617.0	84.0	87.9	132.06	1301.0	0.39	3.84	0.25	85.30	0	0
ESC25Q15max5	9	27	15	8.0	160.0	95.1	98.8	100.00	4.0	0.00	0.00	0.14	0.16	0	0

sible to derive a new approach for solving the DARP. The main modification consists of reducing the paths p in (15) to be feasible according to the time constraints, thus a more elaborated algorithm for finding a shortest path must be applied to solve the separation problem of the Benders' cuts (18). To describe this algorithm it is convenient to compute the values w_i associated to

Table 5
Extended computational results of Class 2 and Alg. 4

			Cuts		Lower bounds				Sep. time					
m	n	Q	SEC	others	SEC	others	opt./PC	B&C	SEC	others	root-t	time	unf.	t.l.
5	11	15	6.6	20.5	96.7	98.0	100.00	4.1	0.00	0.00	0.03	0.07	0	0
5	11	10	6.8	29.6	94.8	97.6	103.32	7.5	0.00	0.01	0.03	0.05	0	0
10	11	20	3.7	7.5	95.0	98.3	102.12	0.6	0.00	0.00	0.00	0.01	0	0
10	11	15	4.2	17.8	90.0	95.2	111.96	1.9	0.00	0.00	0.02	0.02	1	0
10	11	10	3.0	7.7	96.3	100.0	103.02	0.0	0.00	0.00	0.00	0.00	7	0
15	11	500	0.9	6.6	97.4	99.5	100.00	0.6	0.00	0.00	0.00	0.01	0	0
15	11	30	0.6	2.3	99.8	100.0	100.00	0.0	0.00	0.00	0.00	0.01	2	0
15	11	25	0.5	0.5	98.6	100.0	101.51	0.0	0.00	0.00	0.00	0.01	4	0
15	11	20	0.5	1.8	97.9	100.0	102.26	0.0	0.00	0.00	0.00	0.01	6	0
15	11	15	0.0	0.0	100.0	100.0	100.00	0.0	0.00	0.00	0.00	0.01	9	0
15	11	10	0.0	0.0	0.0	0.0	0.00	0.0	0.00	0.00	0.00	0.00	10	0
5	16	15	30.4	247.1	91.8	94.5	100.00	399.1	0.04	0.18	0.03	1.26	0	0
5	16	10	29.9	284.9	90.1	94.0	102.68	363.6	0.02	0.13	0.03	1.21	0	0
10	16	30	24.1	264.0	90.6	94.5	100.00	90.3	0.02	0.06	0.04	0.29	0	0
10	16	25	23.9	270.6	90.5	94.5	100.04	98.7	0.01	0.07	0.04	0.33	0	0
10	16	20	25.9	333.9	89.1	93.7	101.73	123.3	0.02	0.08	0.05	0.50	0	0
10	16	15	26.1	366.8	89.0	93.0	106.34	154.0	0.01	0.10	0.05	0.58	1	0
10	16	10	21.0	580.0	84.6	90.3	108.52	414.0	0.03	0.21	0.07	2.81	7	0
15	16	30	12.0	106.7	93.3	98.0	100.00	9.2	0.00	0.01	0.03	0.05	0	0
15	16	25	12.0	107.1	93.3	98.0	100.01	9.2	0.00	0.01	0.03	0.05	0	0
15	16	20	13.5	312.5	88.7	95.4	105.82	463.1	0.02	0.28	0.05	1.67	2	0
15	16	15	14.0	694.5	78.3	86.4	117.50	866.2	0.07	0.40	0.05	3.87	4	0
15	16	10	0.0	0.0	0.0	0.0	0.00	0.0	0.00	0.00	0.00	0.00	10	0
5	21	15	29.6	169.1	95.9	97.8	100.00	48.9	0.01	0.07	0.04	0.40	0	0
5	21	10	39.7	446.9	93.5	97.2	104.01	386.8	0.05	0.43	0.05	10.51	0	0
10	21	25	36.0	722.6	91.6	96.4	100.00	2102.9	0.21	1.81	0.06	114.68	0	0
10	21	20	39.3	826.5	90.8	96.2	100.86	2127.4	0.20	1.80	0.07	115.06	0	0
10	21	15	50.1	1398.3	87.0	93.2	106.67	3519.9	0.30	2.93	0.11	133.42	0	0
10	21	10	41.8	1476.3	83.6	93.3	108.24	2158.2	0.22	1.89	0.14	112.73	2	2
15	21	500	38.2	1710.9	83.4	89.5	100.00	5140.6	0.44	4.37	0.11	204.61	0	0
15	21	30	37.8	1823.4	82.7	88.8	100.87	5565.1	0.48	4.74	0.10	214.70	0	0
15	21	25	42.8	2798.0	81.5	87.7	102.39	10647.0	1.00	9.46	0.10	649.35	0	0
15	21	20	44.6	1741.0	83.0	89.6	103.52	2240.8	0.20	2.43	0.11	64.58	1	4
15	21	15	39.0	4917.3	79.4	86.5	109.51	10894.0	1.29	10.77	0.17	1230.19	4	3
15	21	10	0.0	0.0	0.0	0.0	0.00	0.0	0.00	0.00	0.00	0.00	10	0
5	26	20	49.4	335.1	94.8	96.4	100.00	177.1	0.04	0.33	0.09	2.77	0	0
5	26	15	57.5	378.2	94.5	96.2	100.26	207.3	0.01	0.40	0.09	3.08	0	0
5	26	10	69.9	642.2	91.8	95.9	103.96	498.0	0.09	0.82	0.10	8.83	0	0
10	26	25	69.0	1179.3	89.4	92.7	100.00	865.4	0.10	1.65	0.20	22.08	0	0
10	26	20	69.7	1424.4	88.8	92.4	100.74	1079.6	0.15	2.09	0.22	30.80	0	0
10	26	15	82.3	2817.9	86.2	91.0	104.12	3680.2	0.43	7.02	0.32	216.71	0	1
10	26	10	55.0	2817.0	86.9	92.4	104.70	3399.0	0.46	6.79	0.38	194.04	1	6
15	26	500	72.0	2845.9	85.5	90.7	100.00	6952.7	0.76	10.80	0.31	328.18	0	1
15	26	30	77.9	3028.7	85.5	90.7	100.00	6132.6	0.75	11.27	0.43	388.59	0	1
15	26	25	66.7	2926.6	85.8	90.5	100.57	3701.9	0.46	7.10	0.35	216.47	0	3
15	26	20	79.8	5080.3	84.2	90.1	102.03	7574.5	1.10	14.51	0.49	1112.70	0	4
15	26	15	50.0	5241.5	84.6	92.1	108.08	2571.0	0.42	5.64	0.88	268.89	4	4
15	26	10	0.0	0.0	0.0	0.0	0.00	0.0	0.00	0.00	0.00	0.00	8	2

Table 6
Extended computational results of Class 3 and Alg. 4

			Cuts		Lower bounds				Sep. time					
m	n	Q	SEC	others	SEC	others	opt./PC	B&C	SEC	others	root-t	time	unf.	t.l.
5	12	20	11.4	47.9	93.2	96.7	100.00	6.4	0.00	0.00	0.01	0.02	0	0
5	12	15	11.4	49.5	93.1	96.7	100.03	6.5	0.00	0.00	0.01	0.03	0	0
5	12	10	13.3	55.3	91.6	96.3	102.70	7.0	0.00	0.00	0.01	0.02	0	0
5	12	5	4.8	31.0	94.5	98.6	122.02	2.6	0.00	0.00	0.01	0.01	0	0
10	22	30	50.8	573.7	89.9	93.0	100.00	364.8	0.04	0.45	0.08	3.75	0	0
10	22	25	51.7	602.7	89.7	92.9	100.31	488.3	0.05	0.55	0.08	4.39	0	0
10	22	20	56.9	774.6	89.2	92.4	100.85	824.6	0.09	1.00	0.08	9.95	0	0
10	22	15	69.5	1992.8	86.0	90.1	105.23	3853.9	0.32	4.03	0.09	155.40	0	0
10	22	10	50.5	3427.1	81.1	91.0	116.01	3795.5	0.53	4.98	0.21	477.83	0	0
10	22	5	23.3	644.7	84.8	95.3	150.47	86.6	0.02	0.19	0.15	1.69	0	0
15	32	30	144.8	3807.3	86.7	90.4	100.00	9183.7	1.14	24.56	0.40	659.33	0	0
15	32	25	144.4	3998.4	86.5	90.3	100.19	7248.2	0.93	20.00	0.41	620.74	0	0
15	32	20	132.1	4688.4	84.7	90.2	103.81	12270.7	1.61	35.67	0.57	1154.04	0	3
15	32	15	122.8	5400.8	84.5	91.1	103.05	5774.5	0.93	18.45	0.96	752.43	1	5
15	32	10	119.0	6343.0	86.7	90.6	109.90	6479.0	1.61	28.81	1.14	1373.31	4	5
15	32	5	32.5	5442.7	80.4	92.7	170.34	2327.7	0.78	10.06	1.26	755.81	1	3

each solution of the master problem x' by solving:

$$\min\{w_{n+1} \ : \ w_0 = 0 \ , \ w_j \geq w_i + t_i + t_{ij}x'_{ij} \ \text{ for all } (i,j) \in A\},$$

which can be done in $O(|A|)$-time by breadth-first search. Then, for each commodity k, a shortest path p' from s_k to t_k must be computed considering the time constraints. This is a folklore problem and the classical way to approach it is by applying dynamic programming.

Acknowledgments

Work partially funded by "Ministerio de Ciencia y Tecnología" (TIC2003-05982-C05-02), Spain.

References

[1] S. Anily and J. Bramel. Approximation algorithms for the capacitated traveling salesman problem with pickups and deliveries. *Naval Research Logistics*, 46:654–670, 1999.

[2] S. Anily, M. Gendreau, and G. Laporte. The swapping problem on a line. *SIAM Journal on Computing*, 29(1):327–335, 1999.

[3] S. Anily and R. Hassin. The swapping problem. *Networks*, 22:419–433, 1992.

[4] N. Ascheuer, M. Jünger, and G. Reinelt. A branch-and-cut algorithm for the asymmetric traveling salesman problem with precedence constraints. *Computational Optimization and Applications*, 17:61–84, 2000.

[5] E. Balas, M. Fischetti, and W. R. Pulleyblank. The precedence-constrained asymmetric traveling salesman polytope. *Mathematical Programming*, 68:241–265, 1995.

[6] L. Bianco, A. Mingozzi, S. Riccardelli, and M. Spadoni. Exact and heuristic procedures for the traveling salesman problem with precedence constraints, based on dynamic programming. *INFOR*, 32:19–32, 1994.

[7] D. Bienstock, S. Chopra, O. Günlük, and C. Y. Tsai. Minimum cost capacity installation for multicommodity network flows. *Mathematical Programming*, 81:177–199, 1998.

[8] L. D. Bodin and T. R. Sexton. The multi-vehicle subscriber dial-a-ride problem. *TIMS Studies in Management Science*, 2:73–86, 1986.

[9] R. Borndörfer, M. Grötschel, F. Klostemeier., and C. Küttner. Telebus berlin: Vehicle scheduling in a dial-a-ride system. In N.H.W. Wilson, editor, *Computer-Aided Transit Scheduling*, volume 471, pages 391–422. Lectures Notes in Economics and Mathematical Systems, Springer, Berlin, 1999.

[10] R. M. Burstall. A heuristic method for a job sequencing problem. *Operational Research Quarterly*, 17:291–304, 1966.

[11] A. Caprara and M. Fischetti. Branch-and-cut algorithms. In M. Dell'Amico, F. Maffioli, and S. Martello, editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 45–64. Wiley, New York, 1997.

[12] P. Chalasani and R. Motwani. Approximating capacitated routing and delivery problems. *SIAM Journal on Computing*, 28:2133–2149, 1999.

[13] T. Christof and A. Löbel. Porta: Polyhedron representation transformation algorithm. http://www.zib.de/Optimization/Software/Porta/, 2003.

[14] J. F. Cordeau. A branch-and-cut algorithm for the dial-a-ride problem. *INFORMS Operations Research*, 2005.

[15] J. F. Cordeau and G. Laporte. The dial-a-ride problem (darp): Variants, modeling issues and algorithm. *4OR - Quarterly Journal of The Belgian, French and Italian Operations Research Societies*, 1:89–101, 2003.

[16] G. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8:101–111, 1960.

[17] J. Desrosiers, Y. Dumas, and F. Soumis. A dynamic programming solution of the large-scale single vehicle dial-a-ride problem with time windows. *American Journal of Mathematical and Management Sciences*, 6:301–325, 1986.

[18] G. N. Frederickson, M. S. Hecht, and C. E. Kim. Approximation algorithms for some routing problems. *SIAM Journal on Computing*, 7:178–193, 1978.

[19] L. Gouveia and P. Pesneau. On extended formulations for the precedence constrained asymmetric traveling salesman problem. *Networks*, 48:77–89, 2006.

[20] D. J. Guan. Routing a vehicle of capacity greater than one. *Discrete Applied Mathematics*, 81:41–57, 1998.

[21] P. Healy and R. Moll. A new extension of local search applied to the dial-a-ride problem. *European Journal of Operational Research*, 83:83–104, 1995.

[22] H. Hernández-Pérez and J. J. Salazar-González. A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Applied Mathematics*, 145:126–139, 2004.

[23] H. Hernández-Pérez and J. J. Salazar-González. Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. *Transportation Science*, 38(2):245–255, 2004.

[24] Y. Dumas M. M. Solomon I. Ioachim, J. Desrosiers. A request clustering algorithm for door-to-door handicapped transportation. *Transportation Science*, 29:63–78, 1995.

[25] J. J. Jaw, A. R. Odoni, H. N. Psaraftis, and N. H. M. Wilson. A heuristic algorithm for the multi-vehicle advance-request dial-a-ride problem with time windows. *Transportation Research Part B*, 20(3):243–257, 1986.

[26] B. Kalantari, A. V. Hill, and S. R. Arora. An algorithm for the traveling salesman problem with pickup and delivery customers. *European Journal of Operational Research*, 22:377–386, 1985.

[27] M. Kubo and H. Kasugai. Heuristic algorithms for the single vehicle dial-a-ride problem. *Journal of the Operations Research Society of Japan*, 33:354–364, 1990.

[28] O. B. Madsen, H. F. Ravn, and J. M. Rygaard. A heuristic algorithm for a dial-a-ride problem with time windows multiple capacities and multiple objectives. *Annals of Operations Research*, 60:193–208, 1995.

[29] G. Mosheiov. The traveling salesman problem with pickup and delivery. *European Journal of Operational Research*, 79:299–310, 1994.

[30] H. Psaraftis. A dinamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. *Transportation Science*, 14:130–154, 1980.

[31] H. Psaraftis. Analysis of an $o(n^2)$ heuristic for the single vehicle many-to-many euclidean dial-a-ride problem. *Transportation Research Part B*, 17:133–145, 1983.

[32] H. Psaraftis. An exact algorithm for the single vehicle many-to-many dial-a-ride problem with time windows. *Transportation Science*, 17:351–357, 1983.

[33] J. Renaud, F. F. Boctor, and G. Laporte. Perturbation heuristics for the pickup and delivery traveling salesman problem. *Computers & Operations Research*, (29):1129–1141, 2002.

[34] J. Renaud, F. F. Boctor, and I. Ouenniche. A heuristic for the pickup and delivery traveling salesman problem. *Computers & Operations Research*, (27):905–916, 2000.

[35] S. Ropke, J. F. Cordeau, and G. Laporte. Models and branch-and-cut algorithm for the pickup and delivery problems with time windows. *Submitted to Networks*, 2005.

[36] S. Ropke and D. Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Submitted to Transportation Science*, 2005.

[37] K. S. Ruland. *The Pickup and Delivery Problem.* PhD thesis, Washington University, 1995.

[38] K. S. Ruland and E. Y. Rodin. The pickup and delivery problem:faces and branch-and-cut algorithm. *Computers & Mathematics, with Applications*, 33:1–13, 1997.

[39] T. R. Sexton and L. D. Bodin. Optimizing single vehicle many-to-many operations with desired delivery times: I scheduling. *Transportation Science*, 19:378–410, 1985.

[40] T. R. Sexton and L. D. Bodin. Optimizing single vehicle many-to-many operations with desired delivery times: Ii routing. *Transportation Science*, 19:411–435, 1985.

[41] M. Sigurd, D. Pisinger, and M. Sig. The pickup and delivery problem with time windows and precedences. *Transportation Science*, 38:197–209, 2004.

[42] P. Toth and D. Vigo. *Fast Local Search Algorithms for the Handicapped Persons Transportation Problem*, pages 677–690. Kluwer, Boston, 1996.

[43] P. Toth and D. Vigo. Heuristic algorithms for the handicapped persons transportation problem. *Transportation Science*, 31:60–71, 1997.

[44] R. Wolfler-Calvo and A. Colorni. An approximation algorithm for the dial-a-ride problem. Working paper, 2002.