Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 151 (2019) 116–123

Procedia Computer Science

www.elsevier.com/locate/procedia

The 10th International Conference on Ambient Systems, Networks and Technologies (ANT)
April 29 - May 2, 2019, Leuven, Belgium

Buffering locations in retail deliveries

MSc. Dmitrii Erkin[a,b], Dr. Elenna Dugundji[a,b,*], Prof. Dr. Ger Koole[a], Prof. Dr. Joaquim Gromicho[a], Prof. Dr. Rob van der Mei[a,b]

[a] *Centrum Wiskunde & Informatica, Science Park 123, 1098 XG Amsterdam, Netherlands*
[b] *Vrije Universiteit Amsterdam, De Boelelaan 1081-1087, 1081 HV Amsterdam, Netherlands*

Abstract

This research helps a European retailer with its daily goods delivery operations. The challenge is the limited capacity of the transportation network in the Amsterdam area. This leads to very strict requirements for the delivery routing plans. In order to respond to this challenge both an analytical model and a numerical method are proposed. This method is capable of generating routing plans given all the necessary constraints including the cost structure. The model takes into account all incurred costs such as mileage, unsatisfied demand and waiting times; they all are weighted according to their relative importance. Furthermore, in order to reduce waiting times, buffering locations within the Amsterdam area are considered where a truck can park and wait until the loading zone at a store becomes available. The numerical method is approximate in nature and is based on the Column Generation technique. This technique allows iterative explorations of the search space by adding new promising one-truck routes (columns). The Regret construction heuristic is applied to generate an initial solution. New promising columns are generated by means of solving the Pricing Sub-problem which takes into account the duals of the Master problem relaxation. The analysis demonstrates that the buffers help to reduce the waiting times incurred by early arrivals without any drop in the total solution costs. Furthermore, a method is proposed to verify the usefulness of different buffering locations in the model.

Keywords: Mathematical Optimization; Linear Programming; Dynamic Programming; Column Generation; Local Search

1. Introduction

A typical day of a European retailer's shipment department includes about 130 deliveries to 82 stores in the Amsterdam city area. These deliveries are carried out by a heterogeneous fleet of trucks which is comprised of vehicles of 3 different types suitable for either large (Euro trailer), medium (City trailer) or small (Bakwagen) volumes of goods. Each shipment case has an associated time window when a truck is expected to arrive at a store location, the type of

* Corresponding author. Tel.: +31 6 514 59 117;
E-mail address: e.r.dugundji@vu.nl

10.1016/j.procs.2019.04.019

goods which should be delivered, and the demand for that type of goods. It is desirable that only one truck is parked at a store at a time. Furthermore, a truck should not arrive before the beginning of the corresponding time window to avoid waiting times which are defined as the sum of both types of delay incurred by either early arrivals to a store or overlapping arrivals/departures at the same store. The costs of a solution, besides waiting times, include traveling distances and unsatisfied demand. Some locations are available around the city where multiple trucks can be parked at the same time moment. These locations are called buffers, and the intention is to use these buffers to allow the trucks to make intermediate stops and wait until the unloading zones become available.

2. Research statement

The research hypothesis is formulated as follows. Utilizing buffering locations in the Amsterdam city area results in the reduction of the waiting time costs of the European retailer's daily goods delivery operations while the total solution costs do not increase.

A number of assumptions are made. To start with, it is assumed that different types of costs have different importance. This assumption significantly affects the resulting shipment plan generated by the model. Next, it is assumed that it is possible to deliver less goods than the total demand by a store. This assumption is made in order to consider cases when partial demand satisfaction can result in lower traveling and/or waiting time costs. It is also assumed that different types of goods have a relative priority for delivery. In other words, if a store has demand for all types of goods then fresh goods are delivered first. In addition to that, it is assumed that a truck travels at the maximal allowed speed for a particular road segment. Regarding the buffering locations, it is assumed that a particular buffer is used only once for a particular one truck trip. It is also assumed that the most reasonable time to arrive at a store from a buffer is in the middle of corresponding time window because it gives more chance for a truck which arrived during a previous time window, and is still parked, to leave this store, so that overlapping arrivals/departures are eliminated. Finally, the length of a driver's shift is taken into account and it is assumed that a trip does not exceed 8 hours.

3. Model

The problem of the optimal fleet route search is formulated using a complete directed graph $G(V, A)$ where V is the set of nodes and A is the set of arcs. V includes the origin depot v_1, the destination depot v_{n+2}, and all the stores and buffers $v_2..v_{n+1}$ en route. The buffers form subset $B \subset V$. A is comprised of directed arcs between graph nodes.

In addition to the graph, there are a number of static parameters such as the set of vehicle types K, the set of vehicle numbers L (the same cardinality is assumed for all vehicle types), the set of time window numbers per day W (the same cardinality is assumed for all locations), and the set of goods types H. Each vehicle type $k \in K$ has maximal capacity Q^k in terms of the number of containers which can be loaded.

Each node $v_i, i \in V\backslash\{1, n+2\}$ has its service time s_i and demand per product type per time window d_i^{wh}. Furthermore, a binary variable, z_i^w, is introduced for each node which indicates the presence of demand. Moreover, each node $v_i, i \in V \setminus \{1, n+2\}$ has a set of time windows when a vehicle should arrive, described by the beginning and ending times, b_i^w and e_i^w respectively. There are no different time windows for different types of goods. In other words, if there are demands for different goods types for a store within a time window then one vehicle will try to deliver all of these goods.

Each arc from node i to node j $(i, j \in V)$ is characterized by its traveling distance c_{1ij} and traveling time t_{ij}. The second parameter can be either deterministic or stochastic. In the latter case t_{ij} is represented by its mean $E(t_{ij})$ and variance $Var(t_{ij})$. Furthermore, the stochastic traveling times are not stationary.

For each ordered pair of stores (i, j), buffer bu is defined such that the sum of traveling times from i to bu and from bu to j is minimal for this pair (i, j). This results in set $NB = (i, j, bu)$ which, for each pair (i, j), has exactly one element corresponding to this pair's nearest buffer.

The set of all feasible routes Ω is defined. Each route $r \in \Omega$ starts at the origin depot, visits a sub-set of stores within their time windows and returns to the destination depot. Given that at most one vehicle can visit a store within each time window, the binary constant α_{ri}^{klw} is introduced which specifies that store i is visited by vehicle l of type k within time window w following route r. Finally, a binary decision variable, x_r, indicates that route r is included in a solution.

For each route $r \in \Omega$ four types of costs are introduced: the traveling costs c_{1r}, the costs c_{2r} of unsatisfied demand, the costs c_{3r} of early arrivals, and the costs c_{4rq} of overlapping arrivals/departures.

The costs c_{1r} of traveling along a route are computed as the sum of all traveling distances c_{1ij} for the arcs which this route is comprised of (1).

$$c_{1r} = \sum_{(i,j)\in r} c_{1ij} \forall r \in \Omega \tag{1}$$

The costs, c_{2i}^{wh}, incurred for not delivering all the required goods of type h to store i within time window w are computed as the difference between demand d_i^{wh} and the actual amount of delivered goods u_i^{wh}. The costs c_{2i}^{w} of not delivering all the required goods to store i within time window w are computed as the sum of c_{2i}^{wh} for all types of goods h. The costs c_{2r} of not satisfying the demand of route r are computed as the sum of c_{2j}^{y} for all locations (y, j) visited while following route r (2).

$$c_{2i}^{wh} = d_i^{wh} - u_i^{wh}, c_{2i}^{w} = \sum_{h\in H} c_{2i}^{wh}, c_{2r} = \sum_{(y,j)\in r} c_{2j}^{y} \forall w \in W, \forall h \in H, \forall i \in V \setminus B \setminus \{1, n+2\} \forall r \in \Omega \tag{2}$$

The costs c_{3j}^{y} of an early arrival at store j within time window y are defined as being greater than the difference between the beginning of time window b_j^y and the time needed to reach this location (j, y) from previous location (i, w) visited while following this route. The time to reach (j, y) is computed as the sum of arrival time a_i^w at (i, w), service time s_i at i and traveling time t_{ij} from i to j. Furthermore, these costs should also be greater than 0 because negative costs (profit) are not incurred. The costs, c_{3r}, of early arrivals along route r are computed as the sum of c_{3j}^{y} for all locations (y, j) visited while following route r (3).

$$c_{3j}^{y} \geq b_j^y - (a_i^w + s_i + t_{ij}), c_{3j}^{y} \geq 0, c_{3r} = \sum_{(y,j)\in r} c_{3j}^{y} \forall r \in \Omega \tag{3}$$

The costs, c_{4rqi}^{w}, of waiting at store i within time window w for any pair of routes r, q are defined as being greater than the difference between departure de_{ri}^{w-1} from store i within the previous time window $w-1$ following route r and arrival a_{qi}^{w} at store i within time window w following route q. In other words, the case is considered when a vehicle on route q has already arrived at the store and is waiting for a vehicle on route r to depart. Furthermore, these costs should also be greater than 0 because negative costs (profit) are not incurred. The costs c_{4rq} of waiting for the pair of routes r, q are computed as the sum of the waiting times costs for all locations visited by these routes (4).

$$c_{4rqi}^{w} \geq de_{ri}^{w-1} - a_{qi}^{w}, c_{4rqi}^{w} \geq 0, c_{4rq} = \sum_{\forall y\in W\setminus\{1\}} \sum_{\forall j\in V\setminus B\setminus\{1,n+2\}} c_{4rqj}^{y} \forall w \in W \setminus \{1\}, \forall i \in V \setminus B \setminus \{1, n+2\}, \forall r, q \in \Omega \tag{4}$$

The last type of cost makes the problem non-linear unless the following is defined:

$$c_{1rq} = \begin{cases} c_{1r} \text{ if } r = q \\ 0 \text{ otherwise} \end{cases} \quad c_{2rq} = \begin{cases} c_{2r} \text{ if } r = q \\ 0 \text{ otherwise} \end{cases} \quad c_{3rq} = \begin{cases} c_{3r} \text{ if } r = q \\ 0 \text{ otherwise} \end{cases} \quad \forall r, q \in \Omega \tag{5}$$

Given these definitions, the objective function is formulated as follows:

$$\min \sum_{r\in\Omega}(\gamma_1 c_{1rr} + \gamma_2 c_{2rr} + \gamma_3 c_{3rr})x_{rr} + \gamma_3 \sum_{r\in\Omega}\sum_{q\in\Omega} c_{4rq}x_{rq} \tag{6}$$

subject to:

$$x_{rq} \in \{0,1\}, x_{rq} = x_{qr}, x_{rq} >= x_{rr} + x_{qq} - 1 \forall r, q \in \Omega \tag{7}$$

$$\sum_{k\in K}\sum_{l\in L}\sum_{r\in\Omega} \alpha_{ri}^{klw} x_{rr} = z_i^w \forall w \in W, \forall i \in V \setminus B \setminus \{1, n+2\} \tag{8}$$

$$0 \le \sum_{r\in\Omega} x_{rr} \le K_{\max} \tag{9}$$

The objective function (6) aims to minimize the sum of the costs for all routes included in a solution. These costs are weighted by non-negative numeric coefficients γ which determine the relative importance of each type of cost. Decision variable x_{rq} represents an ordered pair of routes r, q and has value of 1 for all pairs which are included in a solution (7). Constraint (8) guarantees that all stores which have demand are visited. The total number of routes in a solution should not exceed $K_{\max}$ (9). The above formulation is called the Integer Master Problem (IMP). The IMP takes into account only the inter-route limitations while all the intra-route constrains are incorporated into a procedure which is responsible for creating one-vehicle routes. This procedure is proposed in the next section.

4. Algorithm

4.1. Column Generation

It is technically difficult to explicitly solve the IMP because the number of feasible routes becomes very large even for instances of a moderate size. For a given day in June 2016 there were 82 stores and 125 deliveries so that the upper bound of the number of feasible solutions is about 125! which is a huge number. In reality this number is much smaller because it is unlikely that a vehicle will visit more than 2 stores following one route because its capacity is relatively small compared to demand. Nevertheless, as this number is still large, a special technique is applied to solve this problem: Column Generation [2].

The idea of Column Generation is to build Ω iteratively by adding new promising routes (columns) which could potentially improve the current solution. In order to generate new columns a linear relaxation of the IMP is formulated by replacing (7) with:

$$x_r \ge 0, x_{rq} = x_{qr}, x_{rq} >= x_{rr} + x_{qq} - 1 \forall r, q \in \Omega \tag{10}$$

This relaxation is called the Linear Master Problem (LMP) and it allows the application of the Simplex method to search for new promising columns. The new columns being added to Ω should have the potential to improve the objective function (6). In terms of the Simplex method it means that they should have a negative reduced cost. The reduced cost of a route is computed as follows:

$$\hat{c_r} = c_r - \beta^T a_r - \beta_0 - \beta_{K_{\max}} \forall r \in \Omega \tag{11}$$

$$c_r = c_{1r} + c_{2r} + c_{3r} \forall r \in \Omega \tag{12}$$

$$a_r = \langle a_{ri}^w \rangle, a_{ri}^w = \sum_{k\in K}\sum_{l\in L} \alpha_{ri}^{klw} \forall w \in W, \forall i \in V \setminus B \setminus \{1, n+2\} \forall r \in \Omega \tag{13}$$

In this formulation, c_r is the total cost of route r, β is the vector of dual variables corresponding to (8), a_r is the vector of the same cardinality as β such that each element indicates whether store i is visited within time window w following route r, β_0 and $\beta_{K_{\max}}$ are the dual variables related to both parts of inequality (9) respectively.

Therefore, the search for new promising columns is an optimization problem which is formulated as follows:

$$\begin{aligned} &\min_{r\in\Omega}\hat{c}_r \\ &\text{subject to all constraints related to a route} \end{aligned} \tag{14}$$

This problem is called the Pricing Sub-Problem.

4.2. Dynamic Programming Formulation for the Pricing Sub-Problem

The Pricing Sub-Problem is solved by the Dynamic Programming Algorithm as proposed in [3]. First, the state space at any location $(i, w), i \in V \setminus \{1, n+2\}, w \in W$ is defined as the set of the following parameters:

- k - the type of vehicle k
- a_i^w - the arrival time
- $\hat{c}_i^w$ - the partial reduced cost of arrival
- $\{q_i^{wh} | \forall h \in H\}$ - the amount of goods of type h for which the cargo space is reserved
- S_i^w - the set of locations that have been visited before arriving at (i, w)

Given these definitions, the value function is formulated as follows:

$$\begin{aligned} &V_{iw}(k, a_i^w, \hat{c}_i^w, \{q_i^{wh}|\forall h \in H\}, S_i^w) = \\ &\min_{(j,y)\in(V,W)}\Big(j = (n+2) \wedge \hat{c}_j^y \vee \\ &j \neq (n+2) \wedge \min\Big(V_{jy}(k, a_j^y, \hat{c}_j^y, \{q_j^{yh}|\forall h \in H\}, S_j^y), V_{jy}(k, ab_j^y, \hat{cb}_j^y, \{q_j^{yh}|\forall h \in H\}, SB_j^y)\Big)\Big) \end{aligned} \tag{15}$$

Subject to:

$$a_j^y = \max(b_j^y, a_i^w + s_i + t_{ij}), ab_j^y = \max(\frac{b_j^y + e_j^y}{2}, a_i^w + s_i + t_{i,bu} + t_{bu,j}) \qquad (i, j, bu) \in NB \tag{16}$$

$$\begin{aligned} u_j^{yh} &= \min(d_j^{yh}, \max{}_{o\in H} Q^{ko} - \sum_{o\in H} q_i^{wo} - \mathbb{1}_{h>1} \sum_{o=1..(h-1)} \hat{u}_j^{yo}), \\ \hat{u}_j^{yh} &= \left\lceil \frac{\max{}_{o\in H} Q^{ko}}{Q^{kh}} u_j^{yh} \right\rceil \qquad h = 1..|H| \end{aligned} \tag{17}$$

$$q_j^{yh} = q_i^{wh} + \hat{u}_j^{yh} \qquad \forall h \in H \tag{18}$$

$$\hat{c}_j^y = \hat{c}_i^w + c_{1ij} + c_{2j}^y + c_{3j}^y - \beta_j^y, \hat{cb}_j^y = \hat{c}_i^w + c_{1,i,bu} + c_{1,bu,j} + c_{2j}^y - \beta_j^y \tag{19}$$

$$S_j^y = S_i^w \cup \{(j, y)\}, SB_j^y = S_i^w \cup \{(bu, 1), (j, y)\} \tag{20}$$

Stopping criteria:

$$a_j^y > e_j^y, ab_j^y > e_j^y \tag{21}$$

$$j \neq (n+2) \wedge \sum_{h\in H} u_j^{yh} = 0 \tag{22}$$

$$(j, y) \in S_i^w \tag{23}$$

In this formulation, $V_{wi}(k, a_i^w, \hat{c}_i^w, \{q_i^{wh}|\forall h \in H\}, S_i^w)$ is the minimal reduced cost of reaching destination depot $n+2$ from the current position at store i within time window w. $V_{w1}(k, 0, -\beta_0 - \beta_{K_{\max}}, \{0|\forall h \in H\})$ gives the minimal reduced cost of traveling from origin depot 1 to destination depot $n + 2$ which is the objective of (14).

The equality (16) along with the inequality (21) assure that the current vehicle arrives at store j within time window y but not earlier than it finishes serving previous store i within time window w taking into account traveling time from store i to store j. The amount of goods u_j^{yh} of each type $h \in H$, which is delivered to store j within time window y, and the corresponding amount of cargo space, $\hat{u}_j^{yh}$, which needs to be reserved, are computed by (17). It is assumed that the types of goods are ordered according to their priority $h = 1..|H|$. The reserved capacity of the current vehicle after visiting store j is calculated by (18). The equality (19) gives the partial reduced cost of this route up until reaching store j within time window y. A recursion step is not performed if no goods can be delivered to the next location, either because the current vehicle capacity was exceeded or there is no demand for the delivery of goods (22).

The value function formulation (15) is extended to deal with the buffering locations. In order to do so, term $V_{jy}(k, ab_j^y, \hat{cb}_j^y, \{q_j^{yh}|\forall h \in H\}, SB_j^y)$ and a minimum operator in the third line of (15) are added. This term represents the option of instead of traveling directly from i to j, a truck goes to buffer bu first. The minimum operator indicates that both options are considered, that is to say direct travel from i to j as well as travel via the nearest buffer, bu, and the option is chosen with the minimal total reduced costs. Next, it is (heuristically) assumed that waiting time costs due to overlapping arrivals/departures for the entire solution will be lower if a truck arrives at a store from its nearest buffer not earlier than in the middle of the store's time window. Given this assumption, the arrival time equality (16) is adjusted accordingly. The partial reduced cost equality (19) is also extended for this case to take into account the traveling costs to and from buffer bu, and to exclude the early arrival costs c_{3j}^y which cannot occur in this case.

The value function (15) needs to be solved $|K|$ times but not necessarily to optimality because in practice only a limited number of new promising columns are needed during each Column Generation iteration.

4.3. Regret Construction Heuristic

The Column Generation technique requires a good initial set of promising columns Ω to start with. A regret construction heuristic is applied in order to populate this set. The idea of this heuristic [1] is to extend a route by visiting a location which would cause the biggest regret if this location was not selected as the next destination.

To start with, the last visited store, i, within time window w is defined as location l, and the set of remaining unvisited locations as R. Then, all the possible route extensions $l \to u \to v, \forall v \in R, u \in R \setminus \{v\}$ are evaluated. For each of these extensions the measure of regret is computed, and after that next route location v is chosen which has the highest value of this measure.

Regret is defined as the amount of demand which exceeds the remaining vehicle capacity:

$$\max\left(\sum_{o \in H} d_i^{oh} - (\min{}_{o \in H} Q^{ko} - \sum_{o \in H} q_i^{wo}), 0\right) \forall k \in K \tag{24}$$

4.4. Local Search

If the current best solution is not improved then a local search procedure is applied in order to explore the search space and hopefully escape from the current local optimum. A pair of 'more than one location' routes which belongs to the current best solution is uniformly selected. Then the regret construction heuristic is applied to the combined set of their locations. Resulting new routes are added to Ω. This procedure is applied to at most 100 pairs of the routes.

4.5. Summary

Based on the ideas of the previous subsections, the following algorithm is proposed to solve the model:

1. An initial solution is generated by applying the regret construction heuristic, and then this initial solution is shaken by applying the local search procedure.
2. The LMP is iteratively solved considering the optimal solution of this relaxation as the lower bound. The improvement is defined as the fact of getting a better (smaller) value of the lower bound.

(a) If there are no improvements of the LMP lower bound in 3 consecutive iterations, the IMP is solved, non-basic routes are dropped and the current best solution is shaken by applying the local search procedure. If this shaking gives a better solution then a new iteration is started, otherwise the algorithm stops and reports the best obtained solution.
(b) If there are some improvements in the LMP lower bound, up to 100 new promising routes for each vehicle type are obtained by solving the Pricing Sub-Problem using the current values of LMP duals. If there are no new routes then the algorithm loops back to step (2a).
(c) The Pricing Sub-Problem is defined as being solved if there is no vehicle type for which exactly 100 new promising routes was obtained. If this is the case then the algorithm proceeds to the last iteration solving the LMP and the Pricing Sub-Problem only one more time, and then applying step (2a).

This algorithm gives a heuristic sub-optimal solution.

5. Results

In order to check the research hypothesis a prototype was developed which utilizes Gurobi 8.0 Mixed Integer Linear Solver and Python Programming Language. By means of this prototype, 5 experiments are conducted. The results of these experiments are presented in Table 1. The first row represents the case when only unsatisfied demand is important ($\gamma_2 = 1.0$). In the second row traveling costs are added ($\gamma_1 = 0.1$). The last three rows are related to the experiments where the waiting time costs are considered $\gamma_3 = 0.1$: without the buffering locations; with the buffers which are used before particular stores only; and with the buffers which are shared between all the stores.

Table 1: Experiments summary

	benchmark solution					generated solution				
	$\gamma_1 c_1$	$\gamma_2 c_2$	$\gamma_3 c^y_{3j}$	$\gamma_3 c_{3rq}$	$\sum$	$\gamma_1 c_1$	$\gamma_2 c_2$	$\gamma_3 c^y_{3j}$	$\gamma_3 c_{3rq}$	$\sum$
$\gamma_1 = 0.0, \gamma_2 = 1.0, \gamma_3 = 0.0$	0.00	0.00	0.00	0.00	0.00	0.00	28.00	0.00	0.00	28.00
$\gamma_1 = 0.1, \gamma_2 = 1.0, \gamma_3 = 0.0$	308.73	0.00	0.00	0.00	308.73	304.15	30.00	0.00	0.00	334.15
$\gamma_1 = 0.1, \gamma_2 = 1.0, \gamma_3 = 0.1$ without buffers	308.73	0.00	18.80	6.20	333.73	**308.96**	**39.00**	**18.30**	**0.00**	**366.26**
$\gamma_1 = 0.1, \gamma_2 = 1.0, \gamma_3 = 0.1$ with linked buffers	308.73	0.00	18.80	6.20	333.73	**309.57**	**33.00**	**11.00**	**0.10**	**353.67**
$\gamma_1 = 0.1, \gamma_2 = 1.0, \gamma_3 = 0.1$ with commonly used buffers	308.73	0.00	18.80	6.20	333.73	**310.49**	**26.00**	**4.40**	**0.10**	**340.99**

The use of the linked buffering locations allows a reduction in waiting time costs c_3 from $18.3 + 0.0 = 18.3$ to $11.0 + 0.1 = 11.1$ units while the number of containers which are not delivered decreases from 39 to 33. Furthermore, the utilization of the commonly used buffering locations allows a reduction in waiting time costs c_3 even further to $4.4 + 0.1 = 4.5$ while the number of undelivered containers decreases to 26. The reduction in waiting times is achieved by decreasing early arrival costs c^y_{3j} from 18.3 to 11.0 and 4.4 respectively while the overlapping arrival/departure costs c_{3rq} slightly increase from 0.0 to 0.1. Therefore, the hypothesis is true. Moreover, the commonly used buffers give a larger cost reduction than the buffering locations linked to stores.

The relative importance of the traveling costs, c_1, the demand satisfaction costs, c_2, and the waiting time costs, c_3, are defined as $\gamma_1 = 0.1$€/km, $\gamma_2 = 1.0$€/item and $\gamma_3 = 0.1$€/minute respectively. These definitions allow us to conclude that the utilization of commonly used buffers leads to total cost savings of $366.26 - 340.99 = 25.27$€ per day, which is $(366.26 - 340.99)/366.26 = 6.9\%$ of the total solution costs without using the buffers. Furthermore, the waiting time costs for the case with commonly used buffers are also reduced by $18.3 - 4.5 = 13.8$€ which is $(18.3 - 4.5)/18.3 = 7.5\%$ of the waiting time costs without using the buffers. On the other hand, the traveling costs increase by $310.49 - 308.96 = 1.53$€ (15.3km more kilometers are traveled) which is $(310.49 - 308.96)/308.96 = 0.49\%$.

6. Conclusion

The analysis demonstrates that buffering locations help to reduce the waiting time costs without a drop in the total solution costs. Furthermore, the commonly used buffers have higher solution cost reductions than the linked buffers. In particular, these shared buffers allow savings of 7.5% of the waiting time costs and 6.9% of the total solution costs, compared to the case without buffers. The solution with cost reductions is obtained at the expense of a small increase in traveling costs (0.0049%).

It was observed that there are two clearly separable intervals of the relative importance γ_2 of demand satisfaction: $[0.5, 1.0]$ and $(1.0, 2.0]$. $\gamma_2 > 1.0$€/item corresponds to the case when the delivery costs dominate other types of costs thus there is no way to consider partial delivery. $\gamma_2 \leq 1.0$€/item stands for the case when the delivery costs are comparable to other types of costs so it becomes possible to trade partial demand satisfaction for shorter traveling distances and/or shorter waiting times. Therefore, the smaller relative importance of the demand satisfaction costs leads to the possibility of considering partial delivery.

The implemented model assesses the usefulness of buffering locations in the Amsterdam area. In order to do so, a number of experiments might be conducted for different shipping dates, and the usage of buffers could be monitored. Based on these statistics a more precise inference would be possible to determine which buffers should be established. If a limited budget is available for such a service then the buffers with higher utilization need to be considered first.

Acknowledgements

This research is the output of my internship at Centrum Wiskunde & Informatica (CWI) Amsterdam. I conducted this work for my Master's Thesis as a student in the Business Analytics Programme at the Vrije Universiteit Amsterdam (VU). During these 6 months I studied the practical case of the Vehicle Routing Problem. Together with my colleagues we went from a business case study definition through data discovery and analysis, and created a prototype of a prescriptive mathematical model which is capable of solving a formulated business challenge.

I would like to thank my CWI supervisor Dr. Elenna Dugundji for all the effort she made in providing me with the necessary data from relevant third parties, and supporting me with activities related to gathering business requirements. I would also like to express my gratitude to my VU supervisors Prof. dr. Ger Koole and Prof. dr. Joaquim Gromicho for their valuable recommendations. Last but not least, I would like to thank Rob van der Mei, my host at CWI.

References

[1] Floudas C.A. and Pardalos P.M. (Eds.) (2009). Encyclopedia of Optimization (2nd ed.). New York, NY : Springer, 2009. - 4626 p.

[2] Hu T.C. and Kahng A.B. (2016). Linear and Integer Programming Made Easy (1st ed.). Switzerland, AG: Springer International Publishing, 2016. - 143 p.

[3] Ropke S. and Cordeau J-F. (2009). Branch-and-Cut-and-Price for the Pickup and Delivery Problem with Time Windows. Transportation Science, 43(3), 267-286.