

Synchronizing vans and cargo bikes in a city distribution network

Alexandra Anderluh¹ · Vera C. Hemmelmayr¹ ·
Pamela C. Nolz²

Published online: 24 March 2016

© The Author(s) 2016. This article is published with open access at Springerlink.com

Abstract One of the significant side-effects of growing urbanization is the constantly increasing amount of freight transportation in cities. This is mainly performed by conventional vans and trucks and causes a variety of problems such as road congestion, noise nuisance and pollution. Yet delivering goods to residents is a necessity. Sustainable concepts of city distribution networks are one way of mitigating difficulties of freight services. In this paper we develop a two-echelon city distribution scheme with temporal and spatial synchronization between cargo bikes and vans. The resulting heuristic is based on a greedy randomized adaptive search procedure with path relinking. In our computational experiments we use artificial data as well as real-world data of the city of Vienna. Furthermore we compare three distribution policies. The results show the costs caused by temporal synchronization and can give companies decision-support in planning a sustainable city distribution concept.

Keywords City logistics · Synchronization · GRASP with path relinking · Cargo bikes · Two-echelon VRP

✉ Alexandra Anderluh
alexandra.anderluh@wu.ac.at

Vera C. Hemmelmayr
vera.hemmelmayr@wu.ac.at

Pamela C. Nolz
pamela.nolz@ait.ac.at

¹ WU (Vienna University of Economics and Business), Vienna, Austria

² AIT Austrian Institute of Technology, Vienna, Austria

1 Introduction

The World Health Organization states in its Annual Report 2013 that two of the major challenges in the upcoming years are the aging of the population and growing urbanization, given that 70 % of the world population is forecast to live in cities by 2050 ([World Health Organization—Centre for Health Development 2013](#)). Urbanization can have a great impact on population health and the environment and is therefore a keyword high on the agenda of authorities and the public.

One important aspect of urbanization is the increasing traffic volume in cities, and with it the potential rise in traffic congestion, pollution and noise nuisance. E-commerce and home delivery services are favored distribution channels, required among others by the elderly population or persons with complex daily schedules who cannot easily make their retail purchases on-site.

Thus, accessibility and mobility for citizens and freight need to be improved in urban areas. A means of reducing inefficient polluting traffic within cities is to switch from conventional vehicles to other transport modes when entering urban zones. This of course requires coordination between different stakeholders in order to achieve an optimal and sustainable utilization of resources. Any re-organization of city logistics must be managed in such a way that traffic is reduced and the modal split shifts towards environmentally friendly transportation modes.

Especially in urban areas, there is a huge potential for consolidation and coordination of distribution flows, since typically small amounts of freight need to be delivered to spatially dispersed locations. The aim of the current paper is to look at how to efficiently organize the distribution of goods in cities by consolidating the transport requirements of different stakeholders and using environmentally friendly transport modes in inner-city areas. Reduced traffic volume, efficiently utilized transport resources and the use of green transport modes counteract negative consequences for the environment and the population and enable more livable urban areas in the future.

To identify the main problems for companies concerning city distribution of goods in Vienna we conducted expert interviews with logistics managers from two different sectors: pharmacy wholesale and distributors of vegetable boxes (veg boxes), who distribute not only locally-grown and organic vegetables but also fruits and sometimes even meat and dairy products from the grower or a small co-operative directly to the end consumer ([Brown et al. 2009](#)). We have chosen pharmacy wholesale because this sector requires a relatively high frequency of deliveries during the day. Each of the 316 pharmacies (status 2013) in Vienna is supplied at least twice a day ([Österreichische Apothekerkammer 2014](#)). The interviewed distributors of veg boxes, on the other hand, are looking for new and more sustainable distribution concepts, although they serve their customers—mainly end consumers—only once a week.

Nevertheless, there are a lot of similarities between those two sectors as our expert interviews have shown: Goods from a variety of suppliers are consolidated at a depot at the outskirts of the city and then delivered by small vans to the customers. Customers are often located at poorly accessible parts of the city such as in pedestrian zones or zones with access restrictions. The main difficulties for the companies result from narrow streets and missing parking space, especially in the city center, which often

leads to congested streets. For the companies these circumstances result in vehicles that are not fully loaded, and therefore to inefficient operations. Thus, costs and emissions could be reduced by switching to a more efficient distribution policy.

In this paper we investigate three possible distribution policies to help solve these issues. The first policy reflects the current delivery scheme used by the pharmacies in which customers are supplied directly from the main depot. This corresponds to a classical vehicle routing problem (VRP), in which inner-city areas are supplied by trucks. This can involve difficulties since a lot of cities have already implemented access restrictions for conventional vehicles. In many cities, the city center is a place of historic interest with narrow, congested streets in which additional constraints on delivery apply such as restricted space, accessibility constraints or time windows for loading. Moreover, it is the heart of the city where tourists and inhabitants want to enjoy their time without having to cope with the nuisances of freight traffic.

To tackle these problems we look at a second distribution policy in which customers in the inner-city are supplied by smaller, eco-friendly vehicles. We investigate a two-echelon distribution scheme where vans perform the delivery outside the city center, on the so-called first echelon, and supply the eco-friendly vehicles, that operate on the so-called second echelon. The transfer of load between first and second echelon vehicles is performed on satellite facilities located outside the inner-city area. The downside of this policy is that space is rare in urban areas and hence possible satellite locations are difficult to acquire.

Finally, to overcome this drawback we investigate a third policy that uses mobile satellites without storage facilities, which, for example, could be a parking space. With this policy temporal synchronization of the first and second echelon vehicles at those satellites is necessary.

The contributions of this paper are the following. First, we develop a solution algorithm based on a greedy randomized adaptive search procedure (GRASP) with PR for a two-echelon, multi-trip vehicle routing problem with temporal synchronization. Our algorithm is fast and efficient, and it can easily be used by urban planners. Furthermore, we look at a case study of the city of Vienna and investigate the three different distribution policies explained above.

The paper is organized as follows. Section 2 provides a literature review. Section 3 describes the problem that we study in detail. Section 4 presents the solution method that we develop. Section 5 shows our computational results, while Sect. 6 concludes the paper.

2 Literature review

We consider a two-echelon, multi-trip vehicle routing problem with synchronization. The classic two-echelon vehicle routing problem (2eVRP) was first solved by [Perboli et al. \(2011\)](#). Since then, exact methods ([Jepsen et al. 2013](#); [Baldacci et al. 2013](#); [Santos et al. 2014](#)) and heuristics ([Hemmelmayr et al. 2012](#); [Crainic et al. 2011b](#)) have been proposed. In the 2eVRP studied in these papers, satellites are assumed to have (little) storage space so that temporal synchronization is not necessary. Moreover, multiple trips are not included, and the second echelon vehicles are based at one satellite and return there at the end of the day.

Another aspect of the problem described in this paper is vehicle synchronization, i.e. vehicles of both echelons have to meet at a certain place (spatial synchronization) at a certain point in time (temporal synchronization). A survey of synchronization in vehicle routing is given in [Drexl \(2012\)](#), and a classification of different types of synchronization is provided.

For a detailed review about vehicle routing problems in city logistics we refer the reader to the survey of [Cattaruzza et al. \(2015\)](#). Furthermore, a review of two-echelon routing problems can be found in [Cuda et al. \(2015\)](#).

In the following, we will focus on those papers that consider problem aspects which are similar to those dealt with in this paper. [Crainic et al. \(2009\)](#) study a time-dependent, two-echelon, multi-trip routing problem with synchronization. However, no solution algorithm is proposed, and only promising directions are indicated.

In [Crainic et al. \(2011a\)](#) a modeling framework for tactical planning in city logistics with uncertain demand is developed. They propose a two-stage stochastic programming formulation with recourse. In the first stage the routing of customer demand to the synchronization points is determined. The recourse strategy of the second stage has to solve a synchronized, scheduled, multi-depot, multiple-tour, heterogeneous VRPTW.

[Nguyen et al. \(2013\)](#) propose a tabu search for the time-dependent multi-zone multi-trip vehicle routing problem with time windows. This problem occurs on the second echelon of two-echelon city distribution schemes. The arrival of first-echelon vehicles at synchronization points determines the availability periods in which the second-tier vehicles must arrive at satellites, load and then service customers.

[Grangier et al. \(2014\)](#) study a two-echelon multiple-trip vehicle routing problem with satellite synchronization and time windows. They propose an adaptive large neighborhood search heuristic to solve the problem. Computational experiments are performed on adapted Solomon instances for the VRP. Though their problem is related to ours, they do not include visits to regular customers on the first echelon and they consider time windows at the customer nodes.

3 Problem description

The problem we seek to solve in this paper is a two-echelon routing problem with synchronization, in which the inner-city delivery on the second echelon is performed by cargo bikes. The customers are divided into two groups. There are those located within the city center called bike-customers, and those located outside this area. These latter customers are supplied by vans and hence referred to as van-customers. Moreover, the vans also supply goods to the cargo bikes via so-called satellites. The satellites are transshipment points where vans and cargo bikes can meet. After loading, the cargo bikes perform their delivery and when they have to reload, they move again to a satellite. The city center is defined as the area within the satellites. As we want to avoid vans completely in the city center we also penalize van trips crossing this area by imposing penalty costs for such trips.

Goods from different suppliers arrive at the van depot located on the outskirts of the city. From there, vans perform the delivery to van-customers and to satellites. The satellites are a set of predefined meeting points which do not have storage facilities.

So vans and cargo bikes must meet in a synchronized way at the same time at the same physical satellite, while waiting times for cargo-bikes and vans are minimized. Similar to vans, cargo bikes start and end their tours at the bike depot located in the city center.

The formulation of our VRP is based on the ones by [Grangier et al. \(2014\)](#) and by [Crainic et al. \(2012\)](#). The problem is defined on a graph $G = (V, A)$. The set of nodes V consists of the set of two depot nodes V_d (one as starting point for bikes and one as starting point for vans), the set of n customers V_c , and the set of m satellites V_s .

Based on the topography of the problem the set of customers V_c is split into two parts: All customers located in a circle $C_I \subseteq P_{V_s}$ (the polygon spanned by the satellites $s \in V_s$) are served by cargo bikes. We denote this subset of bike-customers by V_c^b and its complement $V_c \setminus V_c^b$ by V_c^v (the van-customers). Let $n_1 := |V_c^b|$ and $n_2 := |V_c^v|$.

Furthermore, we can denote the bike depot by v_d^b , and the van depot by v_d^v .

As each physical satellite $s \in V_s$ can be used as supply point for each bike-customer $d \in V_c^b$ a set of cloned satellites $\tilde{s} \in \tilde{V}_s$ is created where each physical satellite s is duplicated n_1 times (once for each bike-customer). So, the extended set of vertices can be denoted by $V_E = V_d \cup V_c \cup \tilde{V}_s$.

The set of arcs A consists of all feasible arcs which is described in detail below.

The first level (served by vans) is defined on $G^v = (V^v, A^v)$ with $V^v = \{v_d^v\} \cup V_c^v \cup \tilde{V}_s$ and $A^v = \{(v_d^v, j) \mid j \in V_c^v \cup \tilde{V}_s\} \cup \{(i, j) \mid i \in V_c^v, j \in V^v\} \cup \{(\tilde{s}, j) \mid \tilde{s} \in \tilde{V}_s, j \in V^v\}$.

The second level (served by bikes) is defined on $G^b = (V^b, A^b)$ with $V^b = \{v_d^b\} \cup V_c^b \cup \tilde{V}_s$ and $A^b = \{(v_d^b, \tilde{s}) \mid \tilde{s} \in \tilde{V}_s\} \cup \{(i, j) \mid i \in V_c^b, j \in V^b\} \cup \{(\tilde{s}, j) \mid \tilde{s} \in \tilde{V}_s, j \in V_c^b\}$.

Each vertex $i \in V_E$ has its specific loading or service time $\lambda_i \geq 0$ and each vertex $i \in V_c$ has its specific demand $d_i > 0$.

Furthermore, we have two fleets of vehicles: a fleet of vans (F^v) and a fleet of cargo bikes (F^b) in non-limited number with $F = F^v \cup F^b$.

Each vehicle type $\kappa = \{v, b\}$ has its specific speed s^v/s^b to derive the travel times τ_{ij}^v/τ_{ij}^b from the distances $\delta_{ij}^v/\delta_{ij}^b$ for each arc $(i, j) \in A^v \cup A^b$, if now separate travel time matrix is provided, and its capacity Q^v/Q^b . For the calculation of costs each vehicle type has vehicle cost c_D^v/c_D^b per unit of distance traveled, driver cost c_T^v/c_T^b per unit of time used and fixed cost c_F^v/c_F^b per vehicle used.

To penalize vans crossing the inner area $C_I \subseteq P_{V_s}$ we use a penalty cost matrix with penalty costs p_{ij} for each arc $(i, j) \in A^v$. Please note that p_{ij} is 0, if an arc does not cross C_I .

So c_{ij}^k represents the costs of vehicle k for traversing arc (i, j) .

Furthermore, we define the following decision variables:

The binary variable

$$x_{ij}^k = \begin{cases} 1 & \text{if vehicle } k \text{ travels from node } i \text{ to node } j \\ 0 & \text{otherwise} \end{cases}$$

and the continuous variables

- $w_i^b \geq 0$ specifying the occurring waiting time for a bike at node i
- $w_i^v \geq 0$ specifying the occurring waiting time for a van at node i
- $t_i^k \geq 0$ specifying the arrival time of vehicle k at node i
- $u_i^b \geq 0$ specifying the load of a bike after serving node i
- $u_i^v \geq 0$ specifying the load of a van after serving node i

Furthermore, we use the constant M_u which represents the total demand of all customers.

The maximum route duration is given by t_{max} and the maximum permitted waiting time at any satellite $\tilde{s} \in \tilde{V}_s$ for each vehicle is defined by w_{max} .

The objective function of our model can then be formulated as:

$$\min \sum_{k \in F} \left(\sum_{i \in V_E} \sum_{j \in V_E} c_{ij}^k x_{ij}^k + \sum_{d \in V_d} \sum_{j \in V_E} c_{Fj}^k x_{dj}^k \right) \quad (1)$$

with

$$c_{ij}^k = \left(\tau_{ij}^\kappa + \lambda_j + w_j^\kappa \right) c_T^\kappa + \delta_{ij}^\kappa c_D^\kappa + p_{ij}$$

$$\kappa = \begin{cases} \{v\} & \forall k \in F^v \\ \{b\} & \forall k \in F^b \end{cases}$$

Subject to:

$$\sum_{j \in V_c^v \cup \tilde{V}_s} x_{v_d j}^k = \sum_{j \in V_c^v \cup \tilde{V}_s} x_{j v_d}^k, \forall k \in F^v \quad (2)$$

$$\sum_{j \in V_c^v \cup \tilde{V}_s} x_{v_d j}^k \leq 1, \forall k \in F^v \quad (3)$$

$$\sum_{j \in V_c^b \cup \tilde{V}_s} x_{v_d j}^k = \sum_{j \in V_c^b \cup \tilde{V}_s} x_{j v_d}^k, \forall k \in F^b \quad (4)$$

$$\sum_{j \in V_c^b \cup \tilde{V}_s} x_{v_d j}^k \leq 1, \forall k \in F^b \quad (5)$$

$$\sum_{h \in V_E} x_{hi}^k = \sum_{j \in V_E} x_{ij}^k, \forall i \in V_c, \forall k \in F \quad (6)$$

$$\sum_{k \in F} \sum_{h \in V_E} x_{hi}^k = 1, \forall i \in V_c \quad (7)$$

$$\sum_{i \in V_E} x_{i\tilde{s}}^k = \sum_{j \in V_E} x_{\tilde{s}j}^k, \forall \tilde{s} \in \tilde{V}_s, \forall k \in F \quad (8)$$

$$\sum_{k \in F^b} \sum_{i \in V^b} x_{i\tilde{s}}^k \leq 1, \forall \tilde{s} \in \tilde{V}_s \quad (9)$$

$$\sum_{k \in F^v} \sum_{j \in V^v} x_{j\tilde{s}}^k \leq 1, \forall \tilde{s} \in \tilde{V}_s \quad (10)$$

$$\sum_{k \in F^b} \sum_{i \in V^b} x_{i\tilde{s}}^k = \sum_{k \in F^v} \sum_{j \in V^v} x_{j\tilde{s}}^k, \forall \tilde{s} \in \tilde{V}_s \quad (11)$$

$$u_i^b - d_j - u_j^b \geq (x_{ij}^k - 1) (d_j + u_j^b), \forall k \in F^b, \forall i \in V^b, \forall j \in V_c^b \quad (12)$$

$$u_i^b \leq M_u \left(1 - \sum_{k \in F^b} \sum_{\tilde{s} \in \tilde{V}_s} x_{i\tilde{s}}^k \right), \forall i \in V_c^b \quad (13)$$

$$0 \leq u_i^b \leq Q^b, \forall i \in V^b \quad (14)$$

$$u_i^v - d_j - u_j^v \geq (x_{ij}^k - 1) (d_j + u_j^v), \forall k \in F^v, \forall i \in V_c^v \cup \{v_d^v\}, \forall j \in V_c^v \quad (15)$$

$$u_i^v - u_s^b - u_s^v \geq (x_{i\tilde{s}}^k - 1) (u_s^b + u_j^v), \forall k \in F^v, \forall i \in V_c^v \cup \{v_d^v\}, \forall \tilde{s} \in \tilde{V}_s \quad (16)$$

$$0 \leq u_i^v \leq Q^v, \forall i \in V^v \quad (17)$$

$$\sum_{i \in V^b} \sum_{j \in V^b} (\lambda_i + \tau_{ij}^b + w_i^b) x_{ij}^k \leq t_{max}, \forall k \in F^b \quad (18)$$

$$\sum_{i \in V^v} \sum_{j \in V^v} (\lambda_i + \tau_{ij}^v + w_i^v) x_{ij}^k \leq t_{max}, \forall k \in F^v \quad (19)$$

$$t_{v_d^v}^k = 0, \forall k \in F^v \quad (20)$$

$$\begin{aligned} & t_i^k + \tau_{ij}^b + \lambda_i + w_i^b - t_j^k \\ & \leq (1 - x_{ij}^k) (t_i^k + \tau_{ij}^b + \lambda_i + w_i^b), \forall i, j \in V_E, k \in F^b \end{aligned} \quad (21)$$

$$\begin{aligned} & t_i^k + \tau_{ij}^v + \lambda_i + w_i^v - t_j^k \\ & \leq (1 - x_{ij}^k) (t_i^k + \tau_{ij}^v + \lambda_i + w_i^v), \forall i, j \in V_E, k \in F^v \end{aligned} \quad (22)$$

$$w_{\tilde{s}}^b = \max \left(0; \sum_{k \in F^v} \sum_{i \in V^v} t_{\tilde{s}}^k x_{i\tilde{s}}^k - \sum_{k \in F^b} \sum_{j \in V^b} t_{\tilde{s}}^k x_{j\tilde{s}}^k \right), \forall \tilde{s} \in \tilde{V}_s \quad (23)$$

$$w_{\tilde{s}}^v = \max \left(0; \sum_{k \in F^b} \sum_{i \in V^b} t_{\tilde{s}}^k x_{i\tilde{s}}^k - \sum_{k \in F^v} \sum_{j \in V^v} t_{\tilde{s}}^k x_{j\tilde{s}}^k \right), \forall \tilde{s} \in \tilde{V}_s \quad (24)$$

$$w_{\tilde{s}}^b \leq w_{max}, \forall \tilde{s} \in \tilde{V}_s, k \in F \quad (25)$$

$$w_s^v \leq w_{max}, \forall \tilde{s} \in \tilde{V}_s, k \in F \quad (26)$$

$$x_{ij}^k \in \{0, 1\}, \forall i, j \in V_E, k \in F \quad (27)$$

$$c_{ij}^k, t_i^k, w_i^v, w_i^b, u_i^v, u_i^b \geq 0, \forall i, j \in V_E, k \in F \quad (28)$$

The objective function (1) minimizes the total transportation costs, including variable costs based on distance and time traveled as well as fixed cost for each vehicle in use. Constraints (2) and (3) ensure that each van leaving the van depot also returns to this depot, while constraints (4) and (5) enforce same for bikes. Constraints (6) and (7) guarantee that each customer is visited exactly once by a vehicle. Constraints (8), (9), (10) and (11) ensure that each cloned satellite can be used at most once by a bike or a van and if it is used as supply point by a bike it has to be visited by a van too. Constraint (12) ensures for each bike that the load after visiting a node is equal to the load before minus the demand of the respective node. Constraint (13) ensures that each bike is empty when reaching a satellite. Constraint (14) ensures that the load of a bike at each visited node does not exceed the bike capacity. Constraint (15) ensures for each van that the load after visiting a node is equal to the load before minus the demand of the respective node. Constraint (16) ensures that the load of a bike at a satellite is added as demand for the van which visits this satellite. Constraint (17) ensures that the load of a van at each visited node does not exceed the van capacity. Constraints (18) and (19) restrict the maximum tour duration of a bike and a van, while constraint (20) enforces the start time of vans at the depot to be zero. Constraints (21) and (22) ensure the scheduling of nodes for bikes and vans. Constraint (23) determines the waiting time of a bike at a node as the difference between the arrival time of the van at this node minus the arrival time of the bike at this node, and if the difference is negative the waiting time of a bike at this node is set to zero. Constraint (24) ensures the same issue for the waiting time of a van at a node. Constraints (25) and (26) ensure the synchronization of bikes and vans at cloned satellites within a maximum permitted time span. Eventually, constraint (27) imposes binary values on the decision variable, while constraint (28) ensures non-negative values for all other variables.

The used notation is given in Table 1.

Figure 1 provides a simplified representation of our routing problem which shows the van depot (large rectangle), the bike depot (small rectangle), a number of potential satellites (triangles) and a number of customers (dots). The customers within the inner circle C_I of the satellites are served by cargo bikes (dashed lines) and all other customers are served by vans (solid lines). At a satellite a cargo bike has to meet with a van to reload before continuing on its route. Satellites like the one shown upper left in Fig. 1 are not used for synchronization purposes.

Figure 2 shows the main challenge in our routing problem, namely the spatial and temporal synchronization of cargo bikes and vans. The abscissa in the diagram is the timeline and the ordinate shows the vehicles operating on a certain route. Van 1 starts at the van depot (large rectangle) and services two customers (dots). Then it arrives at the satellite (triangle) which represents a cloned satellite \tilde{s} , where the cargo bike arrives after starting its route at the bike depot (small rectangle). In the meantime van 2 has also started its route from the van depot and services five customers before it arrives at the next satellite which represents also a cloned satellite \tilde{s} and can therefore

Table 1 Notation

<i>Parameters</i>	
V_d	Set of depots
v_d^v/v_d^b	Van depot/bike depot
V_c	Set of customers
V_c^v/V_c^b	Subsets of van-customers/bike-customers
V_s	Set of physical satellites
\tilde{V}_s	Set of cloned satellites
V_E	Extended set of vertices
V^v/V^b	Subsets of nodes which can be visited by a van/bike
n	Number of customers
m	Number of satellites
n_1	Number of bike-customers
n_2	Number of van-customers
C_I	Inner circle of bike-customers
P_{V_s}	Polygon spanned by satellites
F^v/F^b	Fleet of vans/bikes
F	Fleet of all vehicles
κ	Vehicle type: van v /bike b
c_T^v/c_T^b	Driver costs of a van/bike per unit of time traveled/served/waited
c_D^v/c_D^b	Vehicle costs of a van/bike per unit of distance traveled
c_F^v/c_F^b	Fixed costs of a van/bike
p_{ij}	Penalty cost for arc ij used by a van
d_i	Demand of node i
λ_i	Service/loading time at node i
$\delta_{ij}^v/\delta_{ij}^b$	Distance from node i to node j for a van/bike
τ_{ij}^v/τ_{ij}^b	Travel time of a van/bike from node i to node j
s^v/s^b	Speed of a van/bike
Q^v/Q^b	Capacity of a van/bike
t_{max}	Maximum route duration
w_{max}	Maximum waiting time at each satellite node for a vehicle
M_u	Total demand of all customers
<i>Variables</i>	
x_{ij}^k	=1 if vehicle k travels from node i to node j , 0 otherwise
c_{ij}^k	Costs of vehicle k for traversing arc (i, j) plus serving/waiting at node i plus occurring penalty cost
t_i^k	Arrival time of vehicle k at node i
w_i^v/w_i^b	Waiting time of a van/bike at node i
u_i^v/u_i^b	Load of a van/bike after serving node i

Fig. 1 City distribution representation with vans and cargo bikes

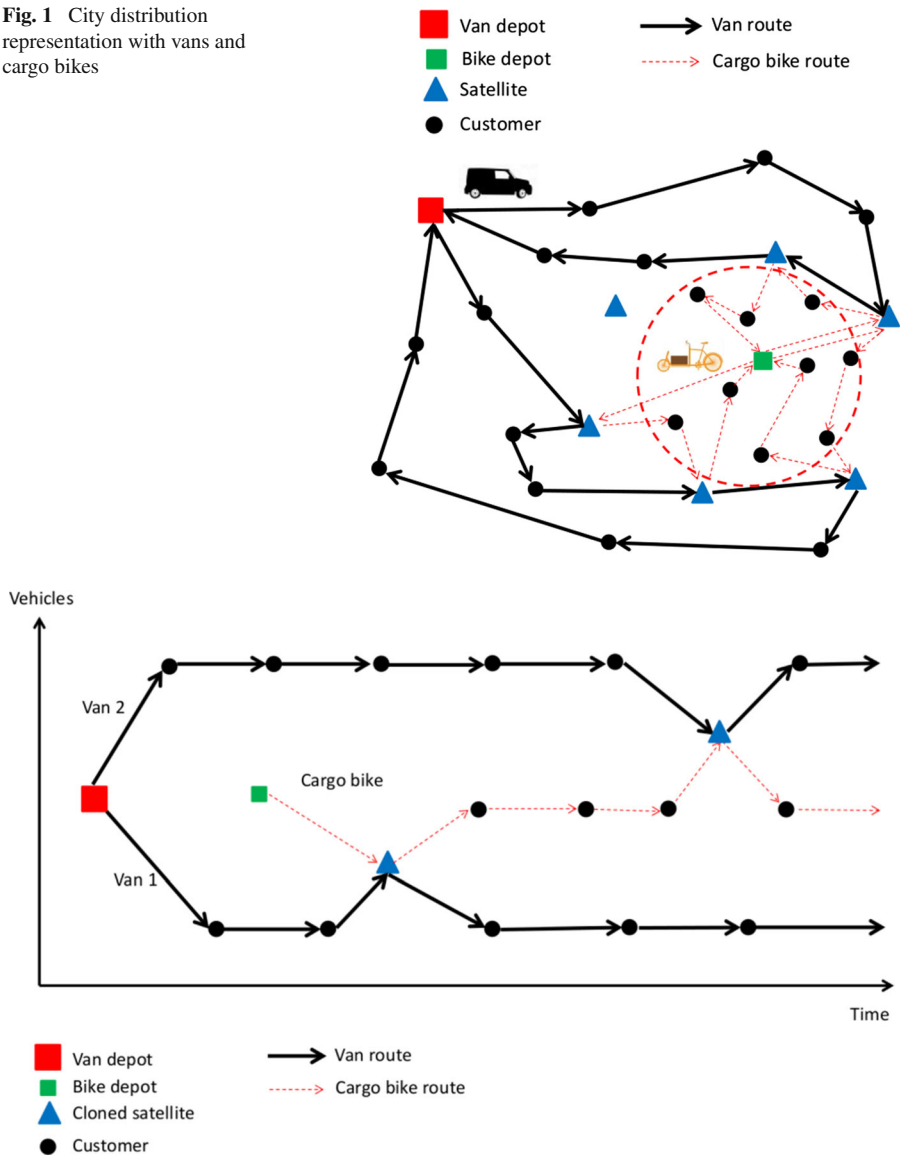


Fig. 2 Spatial and temporal synchronization of cargo bikes and vans

be located at the same physical location than the satellite mentioned above, but it can also be located at another physical satellite location. Here it meets with the cargo bike which has to be reloaded after servicing three customers.

To sum up the potential usage of a physical satellite s we can distinguish the following cases:

1. A physical satellite s is used once in the solution for the meeting of one bike with one van. Then exactly one of the respective cloned satellites \tilde{s} is used in the solution.
2. A physical satellite s is used once in the solution for the meeting of several bikes with one van. Then one respective cloned satellite \tilde{s} for each bike is used in the solution.
3. A physical satellite s is used several times in the solution. Then for each usage in time either case 1. or case 2. can be realized.
4. A physical satellite s is not used at all in the solution, so all respective cloned satellites \tilde{s} stay unused as well.

Our problem is a generalization of the classical VRP and is thus NP-hard.

4 Solution framework

For our city distribution problem with spatial and temporal synchronization between vehicles from and delivery tours to customers on both echelons, we propose a solution procedure which basically consists of GRASP with path relinking (PR). This meta-heuristic is chosen because we search for a fast and easy to use algorithm, which can help to decide, if such a sophisticated distribution scheme can be favorable in contrast to a typical one, where all goods are delivered by one type of vehicle to all customers inside and outside of the city center. Since GRASP is a greedy, memoryless procedure, we combine it with PR. This allows us to combine characteristics of good solutions as part of an intensification strategy.

GRASP was first proposed by [Feo and Resende \(1989\)](#). In general, in each iteration a solution is generated by adding elements from a so-called restricted candidate list (RCL). Usually, the RCL is based on a greedy selection and an element from the list is selected randomly. Another way to combine greediness and random selection is to build the RCL of randomly selected elements and to choose the next element greedily based on costs. Once the solution is generated, it undergoes a local search phase.

The best solution found is stored during the search process and returned at the end (see [Burke and Kendall \(2005\)](#)). GRASP is often used in combination with PR to improve the performance of the algorithm ([Resende and Ribeiro 2005](#)). The initial solution is gradually transformed towards the guiding solution and the solutions on these paths are explored. In PR, the paths leading from one elite solution (ES) to a different one are explored as an intensification strategy. We maintain a solution pool that keeps the best and most diverse solutions. From this pool, two solutions, the initial and the guiding solution, are selected.

In the following subsections we describe our implementation of GRASP with PR.

4.1 Two-phase GRASP

For the construction of van routes on the first echelon we use information on demand and selection of potential satellites given by the bike routes on the second echelon. We thus have a two-phase GRASP, where we start with the construction of cargo bike routes.

Construction of cargo bike routes The initial routes for cargo bikes are constructed by a nearest neighbor heuristic in which the next neighbor is chosen randomly from the RCL.

RCL is a list of all possible (not routed) customers ordered by their insertion costs and is of size α . One customer of this list is either chosen randomly with uniform distribution (unbiased randomization), or randomly with geometric distribution with parameter β (biased randomization), i.e. the probability of being chosen decreases with increasing rank in the list (Juan et al. 2014).

Routes are constructed for each cargo bike successively. The starting node is chosen with (un)biased randomization from a list of satellites ordered by increasing proximity to all remaining bike customers. Then the nearest neighbor heuristic is performed as described above. The bike route is constructed until no more customers can be feasibly added with respect to the route duration constraint. The procedure is then performed for the next bike until all bike-customers are routed.

In order to preserve sufficient capacity and time in the route construction phase we use dummy satellites. This is an efficient way to avoid generating infeasible solutions. In cargo bike routes, we insert a dummy satellite whenever the cargo bike's capacity is exceeded. The cheapest satellite is used as a dummy satellite. At the end of the construction step, we remove all dummy satellites from the routes and perform a local search phase.

In the local search phase we use three operators:

2opt for intra-route changes, *relocate* for inter-route changes of two nodes and *swap* for intra- and inter-route changes of two nodes. This local search is performed for bike routes without any satellites, i.e., before the satellite insertion step.

The *2opt* operator removes the connections between nodes $(i, i + 1)$ and $(j, j + 1)$ and reconnects the two remaining parts of the route, i.e., the new connections are (i, j) and $(i + 1, j + 1)$. As a consequence of this move the route segment between nodes j and $i + 1$ is reversed.

The *relocate* operator removes one node from one route and tries to insert it into another route. Note that this operator can remove the only customer of a route. This empty route is then deleted because it is no longer necessary.

The *swap* operator exchanges a node with another one either located in the same route or in a different route. All inter-route changes only consider routes of the same vehicle type and are based on best improvement. The operators described above are used as long as there is an improvement or the maximum number of local search iterations, θ , is reached.

After the local search phase we insert the satellites in the cargo bike routes with a Dynamic Programming (DP) approach based on the work by Beasley (1983). Based on a bike route consisting of the bike depot node and a number of customer nodes we calculate the insertion costs for the cheapest satellite node \tilde{s} between each pair of nodes. Then we construct a directed graph that consists of the bike depot node as start and end node, and the insertion satellite nodes. Arcs in the graph represent cargo bike trips between two satellite visits. We only generate arcs that reflect the cargo bike's capacity. We can calculate the shortest path in the graph to split the giant tour in feasible cargo bike trips. Whenever we visit a node in this graph, this corresponds to a visit to a satellite \tilde{s} in the original route.

Figure 3 illustrates an example. The upper part shows the bike route starting at the bike depot (small rectangle) which is followed by a number of customers (dots; customer demand is given in brackets) and the bike depot as end node. The triangles in between represent the cheapest insertion satellites \tilde{s} between each pair of nodes (satellite number—for better orientation—and insertion cost are given below each satellite). The lower part of the figure represents the directed graph, where only feasible arcs with respect to the bike's capacity (given as 16) are depicted. The bold line represents the shortest path and indicates the insertion of satellites s_1 , s_2 and s_4 into the bike route to achieve a least costly feasible route.

After this insertion step it may happen that in a route segment between two satellites or the depot and a satellite a detour occurs (see Fig. 4). Therefore, we have added a segmentwise 2opt procedure—*2optseg*—to improve those segments.

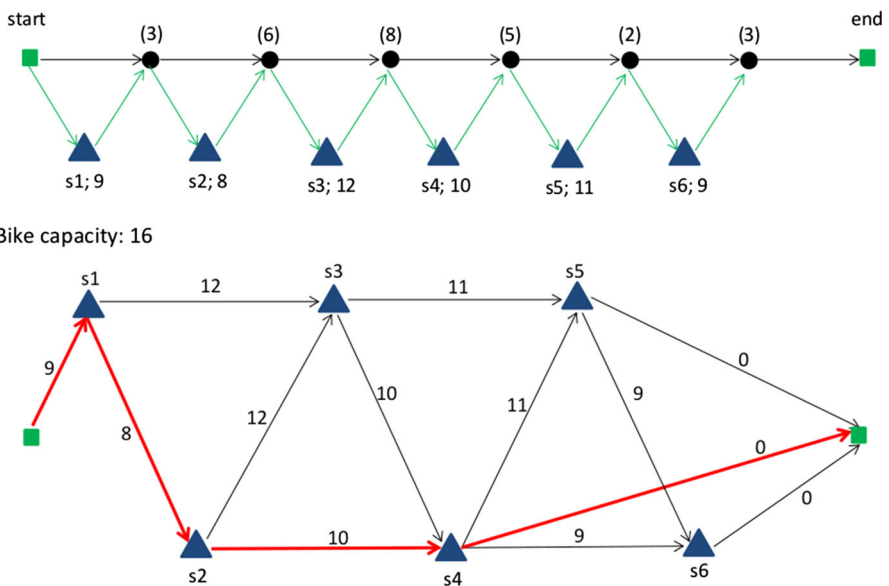
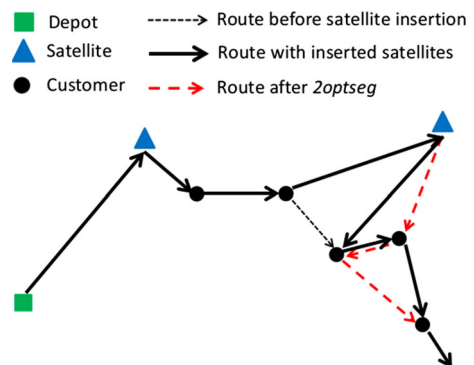


Fig. 3 Dynamic programming example

Fig. 4 Segmentwise 2opt



Construction of van routes Once the cargo bike routes are constructed, we obtain the total demand for each inserted satellite by summing up the demand of all bike-customers on the cargo bike trip originating from that satellite. Moreover, we can calculate the arrival time of the cargo bikes for the respective satellites and use this information for the construction of van routes.

The starting node for all van routes is the depot. The routes are constructed for each van by using the nearest neighbor approach described above. Once the capacity or route duration constraint is about to be exceeded, the van returns back to the depot. For van route construction we also use the concept of dummy satellites. In the initial construction step of a van route all satellites \tilde{s} used by cargo bikes are inserted in a van route without considering the temporal synchronization constraint. This is done to ensure feasible routes. Once all van routes are constructed, the satellites are removed and a local search is performed as described above.

After the construction and improvement steps of the van routes we have to insert all satellites \tilde{s} used by bikes in the van routes with respect to the temporal synchronization constraint. We use a simple best-fit approach, in which we start with the satellite with the earliest scheduled arrival time of a cargo bike. We search for the best (least costly) feasible insertion position in a van route.

At a satellite \tilde{s} during the meeting between a bike and a van waiting times may occur and we can distinguish three cases:

1. The bike and the van arrive at the respective satellite \tilde{s} exactly at the same point in time, then no waiting time is required ($w_s^b = w_s^v = 0$).
2. The bike arrives earlier at satellite \tilde{s} than the van, then the bike has to wait till the arrival of the van and the respective waiting time is assigned to satellite \tilde{s} with $w_s^b = t_s^v - t_s^b \leq w_{max}$.
3. The van arrives earlier at satellite \tilde{s} than the bike, then the van has to wait till the arrival of the bike and the respective waiting time is assigned to satellite \tilde{s} with $w_s^v = t_s^b - t_s^v \leq w_{max}$.

Therefore, in our setting we have a kind of flexible time windows at the satellites \tilde{s} .

Before we can insert the next satellite the *2optseg* is applied to the respective van route where the last satellite has been inserted and then we have to update all concerned bike and van route data with respect to arrival time, demand and overall route duration. Furthermore, if the bike has to wait at the satellite in question all arrival times have to be updated at the successive nodes in the same cargo bike route. If no feasible insertion position can be found for the satellite in question, then a new van route is created which is a mere depot-satellite-depot tour.

This is repeated until all satellites are inserted in a van route and afterwards the feasibility of the solution is checked. Infeasible solutions are discarded.

4.2 PR-path relinking

The GRASP is enhanced with a PR phase. We use PR during GRASP as an intensification strategy and as post-optimization. For PR, we keep a pool of solutions, P , of size ξ . Whether or not a solution from the GRASP phase or after a PR step is accepted to the pool depends on the objective value and the diversity of the solution. A solution s

can only be accepted if the objective function value is of a certain quality with respect to the best solution in the pool, s^* , i.e., $f(s) < f(s^*)(1 + \gamma)$, $\gamma > 0$. The diversity of a solution is measured by comparing the number of bikes used, the number of vans used, the number of synchronizations and the absolute difference of the number of satellite visits for each physical location of a satellite. In order to be accepted to the pool, a solution has to have a minimum difference, ν , to each solution in the pool. A solution that is better than the best solution in the pool is always accepted.

When a new solution is accepted, and the size of the pool becomes larger than the maximum pool size, ξ , another solution has to be removed. In that case, we remove the solution that is most similar to the new one and also that has a cost worse than the new one.

The procedure we use for path generation is based on [Nguyen et al. \(2012\)](#). We use PR on the second echelon, i.e., on the cargo bike routes only. First, the different trips are concatenated, starting with the cargo bike route with the earliest arrival time, to form a giant tour.

The path generation proceeds as follows: For giant tour A , which is a copy of the initiating solution, and giant tour B , which is the guiding solution and remains unchanged, we first make sure that they are of the same length and include the same nodes, i.e., if A has different satellite visits to B , these missing satellites are added and vice versa. Next, we look for a customer with a different position in A than in B . This customer is swapped in A with the node that is on its according position in B . Once there are no customers with different positions, the satellite occurrences are aligned by swaps and substitutions.

Using two example solutions with satellites $s1-s3$ and customers $c1-c5$ the PR procedure looks as follows.

Initial solution A :

s1	c1	c2	c3	s2	c4	c5
----	----	----	----	----	----	----

Guiding solution B :

s3	c4	c2	s2	c1	c3	s1	c5
----	----	----	----	----	----	----	----

As satellite $s3$ only occurs in B we have to add this satellite to A to balance the solutions. This leads to the balanced initial solution A :

s1	c1	c2	c3	s2	c4	c5	s3
----	----	----	----	----	----	----	----

The PR starts with node $c1$ which is the first customer at position 1 in A , but its occurrence in B is at position 4. Therefore we swap $c1$ with $s2$. Proceeding this procedure leads to the following steps:

s1	s2	c2	c3	c1	c4	c5	s3
s1	s2	c2	c4	c1	c3	c5	s3
s1	c4	c2	s2	c1	c3	c5	s3
s1	c4	c2	s2	c1	c3	s3	c5

Now all customers are swapped and we start with the satellites. As $s1$ is not at the correct position we need a swap and we get

$s3$	$c4$	$c2$	$s2$	$c1$	$c3$	$s1$	$c5$
------	------	------	------	------	------	------	------

which is now identical with the guiding solution B .

After each swap we split the newly formed giant tour in single bike tours. Splitting is always done between a customer and a satellite before the maximum route duration is reached. Potentially occurring satellites at the end of a bike route are then discarded. If these bike routes are feasible, the above-mentioned GRASP steps to find a complete solution are added. That is, based on the information provided by the already constructed bike routes we generate then the van routes and insert the satellites as described in Sect. 4.1 with additional respect to the time synchronization constraint. For infeasible bike routes we insert new satellites to achieve feasibility and proceed as described. Feasible solutions are then checked for insertion in the ES pool.

PR is used each time a new GRASP solution is accepted to the non-empty pool. Then PR is performed between this solution and the solution from the pool which is most diverse. The solution with the better objective value acts as the initial solution. PR is also used as a post-optimization step when the pool size is reached. All pairs of solutions from the pool are combined and new solutions added to a new pool. If a new best solution is found, another iteration of post-optimization on this new pool is done and so on [see [Resende and Ribeiro \(2005\)](#) for more details]. Algorithm 1 shows the steps of the overall algorithm.

Algorithm 1 GRASP with PR

```

1:  $f(s^*) \leftarrow \infty$ 
2: repeat
3:    $s' \leftarrow \text{GRASP}$  /* best solution of  $\eta$  GRASP iterations */
4:   if  $\text{AcceptToPool}(s')$  then
5:     if  $|P| \geq \xi$  then
6:       Remove solution which is worse and most similar to  $s'$ 
7:     end if
8:     Add  $s'$  to  $P$ 
9:     Select  $s \in P$  most diverse to  $s'$ 
10:     $s'' \leftarrow \text{argmin}(f(s), f(s'))$ 
11:    repeat
12:       $s'' \leftarrow \text{Perform one move of PR}$ 
13:      if  $\text{AcceptToPool}(s'')$  then
14:        if  $|P| \geq \xi$  then
15:          Remove solution which is worse and most similar to  $s'$ 
16:        end if
17:        Add  $s''$  to  $P$ 
18:      end if
19:    until  $\Delta(s'', s) = 0$ 
20:    end if
21:  until  $|P| = \xi$  /* stopping condition is met */
22:  Post-optimization
23:  return  $\text{argmin}_{s \in P} f(s)$ 

```

5 Computational results

In this section, we present our computational results. First, we give a detailed instance description. Then we show the results for our parameter tests, and compare the performance of GRASP, GRASP with integrated PR, and GRASP with integrated PR and PR post-optimization phase. Finally, we present a comparison of the three different distribution policies described in Sect. 1.

The algorithm was implemented in C++ and tested under Linux Ubuntu 14.04 LTS running on a Virtual Machine (using two processors and 2 GB memory) on a host Intel(R) Core(TM) i5-3320 M CPU @ 2.60 GHz 4 GB RAM.

5.1 Instance description

For testing the performance of our algorithm we use three types of instances.

As first group of instances we use adapted Solomon instances based on those used by [Grangier et al. \(2014\)](#). The following additional changes are necessary:

1. Our van depot [resembling the CDC in [Grangier et al. \(2014\)](#)] is always located at 50/100.
2. The depot in the Solomon instances is used as our bike depot.
3. We use the due time information of the Solomon-depot as our maximum route duration (only for the Solomon RC-instances this value has to be increased to enable feasible solutions).
4. As our two vehicle types drive with different speeds and have different costs we add these relations to the instance. We assume vans to drive three times as fast as cargo bikes. Furthermore vehicle costs for vans are set three times as high as for cargo bikes. The driver costs for van drivers are set 1.1 times higher than the costs for cargo bike drivers.

For more details about the primal adaptations we refer to [Grangier et al. \(2014\)](#). As the instances in each of the six groups (C1, C2, R1, R2, RC1 and RC2) only differ in the time windows, we use only the first instance out of each group (i.e. C101, C201, R101, R201, RC101, RC201) as test instances in our study.

The second group of 12 instances is generated based on the idea of the Solomon instances applied to the city of Vienna. We use 2 clustered, 2 random and 2 combined settings with 100/125 customers and 10 satellites each. Each customer's demand d_i is assumed as a homogeneous good and generated as a random integer between one and eight (i.e., half of the assumed maximum capacity of a cargo bike) as well as each customer's service time t_i^l , which is taken as a random integer between 6 and 16 min. We denote these instances by n100-c1, n100-c2, n100-r1, n100-r2, n100-rc1, n100-rc2, n125-c1, n125-c2, n125-r1, n125-r2, n125-rc1 and n125-rc2.

We also develop a real-world test instance based on the city of Vienna and therefore named 'vienna'. We choose the locations of 100 pharmacies at random and define the van depot as the location of one pharmacy wholesaler. The 18 satellites are placed randomly at potential locations (e.g. parking spaces) around the city center and the bike depot is located at a potential parking place for bikes in the city center (see Fig. 5).

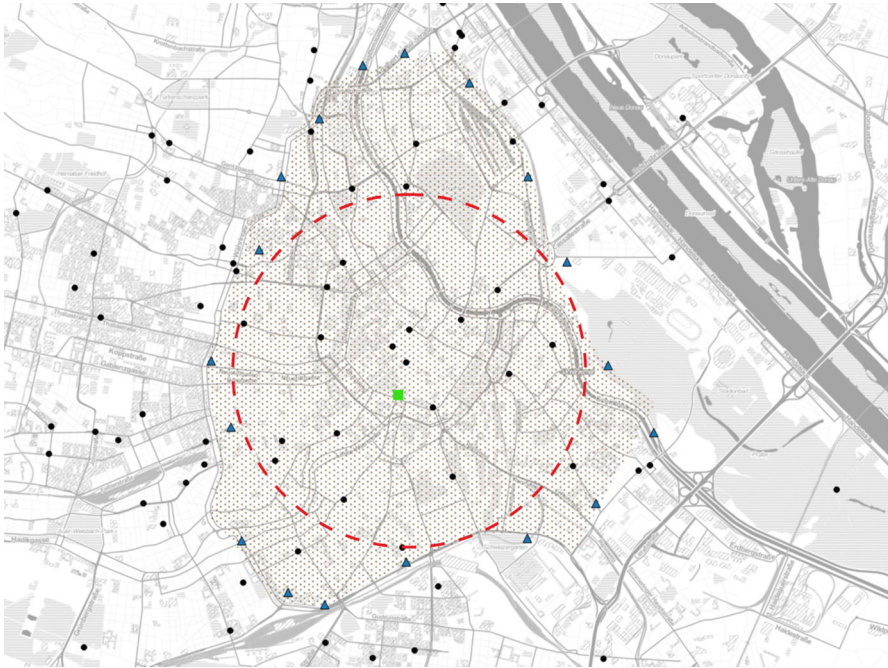


Fig. 5 Selected bike depot (rectangle), satellites (triangles) and customers (dots); dashed circle marks area C_I which should not be crossed by vans; ©OpenStreetMap contributors—www.openstreetmap.org/copyright

Table 2 Assumed values for vehicles for the Vienna instances—all costs in cost units

Vehicle type	Capacity	Vehicle costs	Driver costs	Fixed costs
Van	100 units	0.40 per km	0.33 per min	30 per van
Cargo bike	16 units	0.04 per km	0.30 per min	10 per bike

Instead of euclidean distances we use travel times based on the real street network in Vienna. For the cargo bikes we use an average speed of 15 km/h (Energiregion 2009).

To calculate the travel times for the vans based on the street network we take Floating Car Data (FCD) from FLEET, an FCD system continuously working since 2003 and collecting data from about 3000 taxis in Vienna. Out of this data the system estimates the average speed per road element and time interval (Graser et al. 2012). We use the average speed-per-road element at 4:00 am on a typical weekday to calculate the travel times for the vans.

Service times and demands for each node are assumed as described above. The maximum route duration t_{max} is assumed as 300 min for our test instances and as the duration of the time window of the depot for the Solomon instances. The maximum allowed waiting time w_{max} is assumed as $\frac{t_{max}}{10}$ which holds for all test instances.

For the two homogeneous vehicle fleets we assume the values in Table 2 based on usual vehicle sizes and costs for vehicles and drivers. Furthermore, we use a penalty

cost of 50 units to avoid van trips crossing the city center, which is defined as the largest possible in-circle C_I with its center at the centroid of the polygon P_{V_S} formed by all satellites (see dashed circle in Fig. 5). These penalty cost is also used for the adapted Solomon instances described above.

5.2 Parameter settings

To define the parameters for our algorithms we perform tests for all test instances described in Sect. 5.1 with different parameter combinations (length of RCL α , maximum number of local search iterations θ , parameter of the geometric distribution β , size of the ES pool ξ , and diversity measure ν) and a fixed number of GRASP iterations $\eta = 10$. The cost threshold for accepting solutions to the ES pool γ is set to 1 in general to guarantee a sufficient number of diverse solutions.

Then we rank the results with respect to average total costs (column costs) in cost units and average computational time (column time) in seconds (see Table 3). Weighing both rankings with 80 % share of cost ranking and 20 % share of time ranking yields PC3/PC9/PC12 for GRASP, PC12/PC9/PC5 for GRASP+PR(I) and PC9/PC12/PC6 for GRASP+PR(I+P) as first/second/third place respectively. Based on these results we decide to use PC9 for all following tests in all settings (see Table 4).

For measuring the influence of runtime on the performance of each algorithm we perform tests with increasing number of iterations η . The maximum runtime of each test is limited with 300s. The results as average values for all instances are shown in Fig. 6. This time limit of 300s represents approximately $\eta = 3000$ for GRASP, $\eta = 300$ for GRASP+PR(I) and $\eta = 190$ for GRASP+PR(I+P). Based on the convergence behavior of our algorithms we decide to use a maximum of $\eta = 500, 200, 100$ for GRASP, GRASP+PR(I) and GRASP+PR(I+P) respectively and limit the runtime additionally at 300s.

5.3 Results for testing GRASP and PR components

We have solved all instances using

- GRASP alone,
- GRASP+PR(I), where the PR step is used as integrated step, and
- GRASP+PR(I+P), where the PR step is also used as a post-optimization phase.

All instances have been solved five times and the average value for each instance is presented in all tables. Table 5 depicts the aggregated total costs for the all instances. Values in brackets show the improvement of each method compared to the previous column.

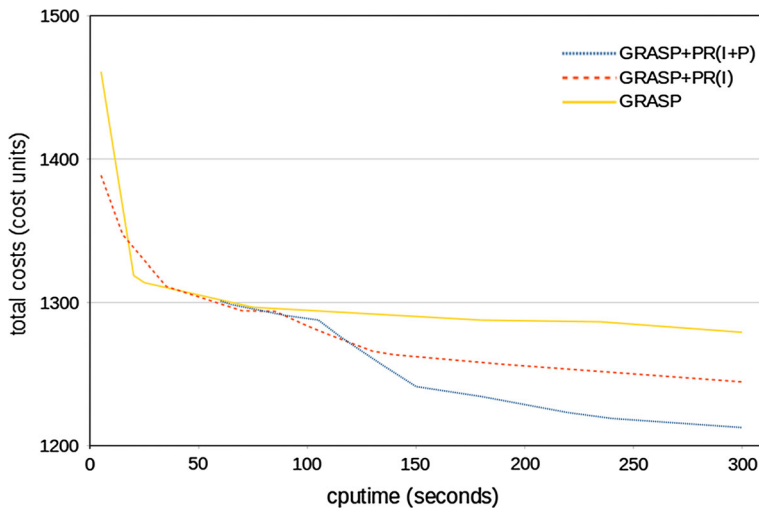
Detailed results for all instances can be found in Tables 6, 7 and 8. Here the number of used bikes/vans ($\#b/v$), the distance traveled by all bikes/vans (dist b/v), the occurring waiting time for bikes/vans (wait b/v), costs for bikes/vans (cost b/v), the sum of all costs ($\sum \text{cost}$), as well as the number of used synchronization points ($\#\text{sync}$) is depicted.

Table 3 Testing of parameter combinations for GRASP, GRASP+PR(I) and GRASP+PR(I+P)

Parameter combination	α	θ	β	ξ	ν	Costs	Time	Rank
<i>GRASP</i>								
PC1	5	10	0.2	5	1	1503.20	1.33	5
PC2	5	10	0.2	7	1	–	–	–
PC3	5	10	0.4	3	3	1385.09	1.07	1
PC4	5	20	0.2	7	1	1505.10	1.30	6
PC5	5	20	0.4	3	3	–	–	–
PC6	5	20	0.4	5	2	1401.08	1.10	4
PC7	10	10	0.2	3	3	1506.99	1.24	7
PC8	10	10	0.2	7	1	–	–	–
PC9	10	10	0.4	5	2	1397.24	1.13	2
PC10	10	20	0.2	5	1	1512.65	1.27	8
PC11	10	20	0.4	3	3	–	–	–
PC12	10	20	0.4	7	1	1399.79	1.14	3
<i>GRASP+PR(I)</i>								
PC1	5	10	0.2	5	1	1438.18	11.04	9
PC2	5	10	0.2	7	1	1433.35	16.26	9
PC3	5	10	0.4	3	3	1351.21	5.25	5
PC4	5	20	0.2	7	1	1427.06	16.91	8
PC5	5	20	0.4	3	3	1347.17	5.80	3
PC6	5	20	0.4	5	2	1338.22	11.09	4
PC7	10	10	0.2	3	3	1454.08	5.31	12
PC8	10	10	0.2	7	1	1425.16	19.16	7
PC9	10	10	0.4	5	2	1333.65	11.60	2
PC10	10	20	0.2	5	1	1443.16	9.12	11
PC11	10	20	0.4	3	3	1355.33	6.44	6
PC12	10	20	0.4	7	1	1326.68	16.03	1
<i>GRASP+PR(I+P)</i>								
PC1	5	10	0.2	5	1	1396.77	77.32	9
PC2	5	10	0.2	7	1	1387.98	140.09	10
PC3	5	10	0.4	3	3	1333.58	15.66	6
PC4	5	20	0.2	7	1	1383.65	104.05	8
PC5	5	20	0.4	3	3	1325.03	17.96	4
PC6	5	20	0.4	5	2	1306.29	61.28	3
PC7	10	10	0.2	3	3	1413.70	22.14	12
PC8	10	10	0.2	7	1	1382.12	109.92	7
PC9	10	10	0.4	5	2	1304.81	63.80	1
PC10	10	20	0.2	5	1	1411.16	67.03	11
PC11	10	20	0.4	3	3	1326.94	20.79	5
PC12	10	20	0.4	7	1	1293.84	122.64	2

Table 4 Parameter setting

Parameter	Value
Length of the RCL α	10
Maximum number of local search iterations θ	10
Parameter of the geometric distribution β	0.4
Size of the ES pool ξ	5
Cost threshold for accepting solutions to the ES pool γ	1
Diversity measure ν	2

**Fig. 6** Runtime versus performance of GRASP, GRASP+PR(I) and GRASP+PR(I+P)

Our computational results illustrate that for most of the instances with randomly assigned customers the PR post-optimization step does not lead to an improvement at all, while in general, instances with clustered customers can be further improved by this step. Besides, the more synchronizations occur in each bike tour, the better the PR step performs which can be seen for example at the results of instance Vienna, where 2 bikes are used on average and a total of 5 synchronizations is necessary. Here the PR(I) step can improve the result by about 3 % and the post-optimization step can add about another 7 % improvement to the result (see Tables 6, 7 and 8 as well as Table 5). If each bike tour requires only one synchronization in a solution, the improvement gained by the PR step is on average rather small (e.g. instances R201 and RC201).

5.4 Comparison of different distribution policies

We investigate three different distribution policies. The first one resembles the current distribution policy classically used by companies which is the solution of a capacitated VRP. In the second one, the use of intermediate satellite facilities is possible

Table 5 Aggregated results for testing PR components

Instance	GRASP	GRASP+PR(I)	GRASP+PR(I+P)
C101	2318.56	2306.30 (0.53 %)	2306.30 (0.00 %)
C201	1694.22	1693.14 (0.06 %)	1693.14 (0.00 %)
R101	1021.53	1014.78 (0.66 %)	1008.75 (0.59 %)
R201	718.23	716.30 (0.27 %)	716.12 (0.03 %)
RC101	982.78	972.64 (1.03 %)	972.64 (0.00,%)
RC201	747.27	741.68 (0.75 %)	741.68 (0.00 %)
n100-c1	1260.61	1244.80 (1.25 %)	1244.80 (0.00 %)
n100-c2	1326.42	1296.28 (2.27 %)	1296.28 (0.00 %)
n100-r1	1179.55	1177.72 (0.16 %)	1177.72 (0.00 %)
n100-r2	1100.40	1093.35 (0.64 %)	1093.35 (0.00 %)
n100-rc1	1306.49	1270.77 (2.73 %)	1270.77 (0.00 %)
n100-rc2	1194.33	1187.72 (0.55 %)	1187.72 (0.00 %)
n125-c1	1783.60	1731.52 (2.92 %)	1645.58 (4.96 %)
n125-c2	1488.32	1453.67 (2.33 %)	1453.67 (0.00 %)
n125-r1	1330.03	1288.17 (3.15 %)	1288.17 (0.00 %)
n125-r2	1339.91	1337.52 (0.18 %)	1337.52 (0.00 %)
n125-rc1	1404.67	1400.02 (0.33 %)	1400.02 (0.00 %)
n125-rc2	1406.39	1397.25 (0.65 %)	1361.12 (2.59 %)
Vienna	934.85	917.52 (1.85 %)	906.56 (1.19 %)

without additional costs. These satellite facilities are assumed to have storage space, so that a temporal synchronization between the first and second echelons is not necessary. Finally, the last distribution policy is the two-echelon distribution scheme with temporal synchronization of vans and cargo bikes investigated in detail in this paper.

As our PR builds on bike routes, we use only GRASP to solve the first distribution policy, where only vans are used. The other two distribution policies are solved by GRASP+PR(I+P).

Table 9 presents the aggregated results for the three distribution policies, where numbers in brackets refer again to the increase (+) or decrease (−) in costs as a percentage value compared to the previous column. Please note that we do not consider storage cost for the second distribution policy. Detailed results can be found in Tables 10, 11 and 12 depicting the same columns as in Tables 6, 7 and 8.

Table 9 shows that in instance R201 the combined usage of cargo bikes and vans (column labeled 'without sync') is less costly than the usage of vans alone, although for the solution with vans alone we do not even use a penalty cost for van trips crossing the city center. This can be explained by the fact that in this instance one van can be substituted by two bikes (see Tables 10 and 11) and the underlying cost structure (see Table 2).

For all other instances the combined usage of cargo bikes and vans causes higher costs, but since some vans can be substituted by cargo bikes, a significant reduction

Table 6 Detailed results for GRASP

Instance	# <i>b</i>	# <i>v</i>	dist <i>b</i>	dist <i>v</i>	wait <i>b</i>	wait <i>v</i>	cost <i>b</i>	cost <i>v</i>	\sum cost	#sync
C101	5.00	8.00	813.67	1271.32	12.04	4.25	569.99	1748.57	2318.56	5.60
C201	2.00	3.00	435.75	734.89	21.45	141.64	417.66	1276.57	1694.22	3.60
R101	5.00	3.00	811.36	802.96	0.00	5.86	362.97	658.55	1021.53	5.20
R201	2.00	2.00	409.67	635.65	0.00	0.00	192.38	525.85	718.23	2.00
RC101	4.00	3.00	666.68	838.18	0.00	4.15	296.34	686.45	982.78	4.00
RC201	2.00	2.00	398.33	688.78	0.00	0.00	182.41	564.86	747.27	2.00
n100-c1	2.40	8.00	57.94	446.73	6.00	13.39	191.35	1069.26	1260.61	5.80
n100-c2	3.00	8.00	70.41	470.15	23.76	17.51	224.16	1102.26	1326.42	6.80
n100-r1	2.00	8.00	37.96	403.34	12.94	6.44	121.95	1057.60	1179.55	4.00
n100-r2	1.00	8.00	23.07	380.29	6.00	1.93	77.33	1023.06	1100.40	2.00
n100-rc1	2.00	9.00	39.22	479.89	11.44	9.07	123.43	1183.06	1306.49	4.00
n100-rc2	2.00	8.00	49.98	407.62	12.92	9.37	158.45	1035.88	1194.33	4.60
n125-c1	3.40	11.00	75.75	590.27	18.79	37.33	259.79	1523.81	1783.60	9.80
n125-c2	3.00	9.40	73.98	492.34	11.01	25.18	235.43	1252.89	1488.32	7.00
n125-r1	1.20	10.00	32.24	420.82	7.34	2.33	95.66	1234.37	1330.03	2.00
n125-r2	2.00	9.00	43.93	432.31	8.98	5.22	154.55	1185.36	1339.91	4.00
n125-rc1	2.00	10.00	43.38	482.93	5.18	0.40	129.76	1274.91	1404.67	3.00
n125-rc2	1.20	10.00	36.07	509.01	1.03	16.59	103.85	1302.53	1406.39	4.00
Vienna	2.00	6.00	42.53	278.03	12.54	3.02	121.58	813.27	934.85	5.60

Table 7 Detailed results for GRASP+PR(l)

Instance	# <i>b</i>	# <i>v</i>	dist <i>b</i>	dist <i>v</i>	wait <i>b</i>	wait <i>v</i>	cost <i>b</i>	cost <i>v</i>	∑ cost	#sync
C101	5.00	7.80	839.71	1247.25	8.62	1.02	578.76	1727.54	2306.30	5.40
C201	2.00	3.00	433.11	753.29	26.85	33.28	417.27	1275.87	1693.14	3.40
R101	4.60	3.00	776.38	815.31	7.86	16.60	347.52	667.27	1014.78	5.20
R201	2.00	2.00	408.68	633.06	0.00	0.00	192.04	524.27	716.30	2.00
RC101	4.00	3.00	660.51	824.40	0.00	7.96	294.18	678.46	972.64	4.00
RC201	2.00	2.00	394.39	681.87	0.00	0.00	181.04	560.64	741.68	2.00
n100-c1	2.20	8.00	57.81	433.29	5.41	19.16	188.19	1056.61	1244.80	5.60
n100-c2	3.00	8.00	74.09	443.20	18.56	4.56	227.07	1069.21	1296.28	6.40
n100-r1	2.00	8.00	41.15	396.05	16.68	5.14	127.97	1049.75	1177.72	4.00
n100-r2	1.00	8.00	23.07	374.22	2.58	2.40	76.31	1017.04	1093.35	2.00
n100-rc1	2.00	9.00	39.22	456.94	8.06	5.03	122.42	1148.35	1270.77	4.00
n100-rc2	2.00	8.00	43.59	418.19	4.78	5.43	144.36	1043.37	1187.72	4.00
n125-c1	3.20	10.40	78.65	574.78	20.99	41.20	263.52	1468.00	1731.52	10.00
n125-c2	3.00	9.00	73.25	478.94	8.61	15.32	232.99	1220.68	1453.67	6.80
n125-r2	1.20	9.20	31.81	410.91	0.97	2.59	93.08	1195.09	1288.18	2.00
n125-r2	2.00	9.00	47.04	420.83	16.86	7.96	162.30	1175.23	1337.52	4.20
n125-rc1	2.00	10.00	43.38	478.69	0.00	4.15	128.20	1271.82	1400.02	3.00
n125-rc2	1.40	10.00	36.82	495.79	2.16	19.12	107.36	1289.90	1397.25	4.00
Vienna	2.00	6.00	51.28	277.31	6.76	21.11	123.27	794.25	917.52	5.00

Table 8 Detailed results for GRASP+PR(t+P)

Instance	# <i>b</i>	# <i>v</i>	dist <i>b</i>	dist <i>v</i>	wait <i>b</i>	wait <i>v</i>	cost <i>b</i>	cost <i>v</i>	\sum cost	#sync
C101	5.00	7.80	839.71	1247.25	8.62	1.02	578.76	1727.53	2306.30	5.40
C201	2.00	3.00	433.11	753.29	26.85	33.28	417.27	1275.87	1693.14	3.40
R101	4.40	3.00	761.14	811.91	14.20	41.51	340.82	667.93	1008.75	5.20
R201	2.00	2.00	407.52	633.43	0.00	0.00	191.63	524.49	716.12	2.00
RC101	4.00	3.00	660.51	824.40	0.00	7.96	294.18	678.46	972.64	4.00
RC201	2.00	2.00	394.39	681.87	0.00	0.00	181.04	560.64	741.68	2.00
n100-c1	2.20	8.00	57.81	433.29	5.41	19.16	188.19	1056.61	1244.80	5.60
n100-c2	3.00	8.00	74.09	443.20	18.56	4.56	227.07	1069.21	1296.28	6.40
n100-r1	2.00	8.00	41.15	396.05	16.68	5.14	127.97	1049.75	1177.72	4.00
n100-r2	1.00	8.00	23.07	374.22	2.58	2.40	76.31	1017.04	1093.35	2.00
n100-rc1	2.00	9.00	39.22	456.94	8.06	5.03	122.42	1148.35	1270.77	4.00
n100-rc2	2.00	8.00	43.59	418.19	4.78	5.43	144.36	1043.37	1187.72	4.00
n125-c1	3.20	10.20	84.32	556.32	14.50	28.83	269.11	1376.47	1645.58	9.60
n125-c2	3.00	9.00	73.25	478.94	8.61	15.32	232.99	1220.68	1453.67	6.80
n125-r2	1.20	9.20	31.81	410.91	0.97	2.59	93.08	1195.09	1288.18	2.00
n125-r2	2.00	9.00	47.04	420.83	16.86	7.96	162.30	1175.22	1337.52	4.20
n125-rc1	2.00	10.00	43.38	478.69	0.00	4.15	128.20	1271.82	1400.02	3.00
n125-rc2	1.20	9.80	35.45	477.94	5.60	11.52	101.88	1259.24	1361.12	3.20
Vienna	2.00	6.00	54.80	263.18	7.61	4.17	126.86	779.69	906.56	5.20

Table 9 Aggregated results for different distribution policies

Instance	Vans only	Without sync	With sync
C101	2067.37	2282.34 (10.40 %)	2306.30 (1.05 %)
C201	1609.02	1642.40 (2.07 %)	1693.14 (3.09 %)
R101	808.49	1003.11 (24.07 %)	1008.75 (0.56 %)
R201	721.06	712.84 (−1.14 %)	716.12 (0.46 %)
RC101	805.11	965.65 (19.94 %)	972.64 (0.72 %)
RC201	704.65	734.12 (4.18 %)	741.68 (1.03 %)
n100-c1	1052.06	1212.86 (15.28 %)	1244.80 (2.63 %)
n100-c2	1104.97	1247.07 (12.86 %)	1302.81 (4.47 %)
n100-r1	1067.22	1159.75 (8.67 %)	1177.72 (1.55 %)
n100-r2	1037.98	1089.25 (4.94 %)	1093.35 (0.38 %)
n100-rc1	1135.64	1178.72 (3.79 %)	1270.77 (7.81 %)
n100-rc2	1046.20	1149.35 (9.86 %)	1187.72 (3.34 %)
n125-c1	1327.99	1553.08 (16.95 %)	1645.58 (5.96 %)
n125-c2	1294.91	1425.68 (10.10 %)	1459.37 (2.36 %)
n125-r2	1241.75	1275.21 (2.69 %)	1288.17 (1.02 %)
n125-r2	1255.96	1314.33 (4.65 %)	1337.52 (1.76 %)
n125-rc1	1284.96	1373.37 (6.88 %)	1400.02 (1.94 %)
n125-rc2	1258.99	1353.60 (7.51 %)	1361.12 (0.56 %)
Vienna	763.65	889.09 (16.43 %)	906.56 (1.96 %)

of CO_2 -emissions can be provided and the presence of vans in the city center can be reduced.

Policies with temporal synchronization (column labeled 'with sync' in Table 9) always lead to a cost increase compared to policies without synchronization. The level of cost increase for the synchronization step depends mainly on the number of required temporal synchronizations compared to the number of required bikes (see Tables 10, 11 and 12). These synchronizations can lead to additional waiting times of vans and/or cargo bikes if the required synchronizations do not only occur at the beginning of a bike route but also in between (i.e. a bike meets with a van not only after leaving the bike depot but also afterwards as in instance n125-c1, where about 3 bikes require about 9 synchronizations). Besides, it can also happen that an additional vehicle is required to achieve a feasible solution as it can be seen for example at instance n100-rc1, where instead of 8 vans now 9 vans are necessary to achieve a feasible solution in case of temporal synchronization. However, given that we do not consider the costs of storage facilities in the 'without sync' policy of Table 9, the additional cost of synchronization are quite low on average with 3.15 %, although those values differ depending on the instance in the range of 0.23 to about 10 %. Comparing the additional costs for synchronization to the storage costs can help planners decide which distribution policy is more preferable.

Figure 7 shows a sample solution for our Vienna test instance where six vans are used to deliver all occurring demand assuming that there are no restricted zones

Table 10 Detailed results for distribution policy with vans only

Instance	# <i>b</i>	# <i>v</i>	dist <i>b</i>	dist <i>v</i>	wait <i>b</i>	wait <i>v</i>	cost <i>b</i>	cost <i>v</i>	∑ cost	#sync
C101	—	9.00	—	1323.55	—	—	—	2067.37	2067.37	—
C201	—	4.00	—	818.07	—	—	—	1609.02	1609.02	—
R101	—	4.00	—	948.34	—	—	—	808.49	808.49	—
R201	—	3.00	—	854.20	—	—	—	721.06	721.06	—
RC101	—	4.00	—	942.80	—	—	—	805.11	805.11	—
RC201	—	3.00	—	827.29	—	—	—	704.65	704.65	—
n100-c1	—	8.00	—	400.62	—	—	—	1052.06	1052.06	—
n100-c2	—	8.40	—	423.30	—	—	—	1104.97	1104.97	—
n100-r1	—	8.20	—	390.84	—	—	—	1067.22	1067.22	—
n100-r2	—	8.00	—	376.12	—	—	—	1037.98	1037.98	—
n100-rc1	—	9.00	—	429.26	—	—	—	1135.65	1135.65	—
n100-rc2	—	8.00	—	388.39	—	—	—	1046.20	1046.20	—
n125-c1	—	10.00	—	495.97	—	—	—	1327.99	1327.99	—
n125-c2	—	10.00	—	470.30	—	—	—	1294.91	1294.91	—
n125-r2	—	10.00	—	407.43	—	—	—	1241.75	1241.75	—
n125-r2	—	10.00	—	423.97	—	—	—	1255.96	1255.96	—
n125-rc1	—	10.00	—	467.66	—	—	—	1284.96	1284.96	—
n125-rc2	—	10.00	—	458.69	—	—	—	1258.99	1258.99	—
Vienna	—	6.00	—	233.36	—	—	—	763.65	763.65	—

Table 11 Detailed results for distribution policy without synchronization between vans and cargo bikes

Instance	# <i>b</i>	# <i>v</i>	dist <i>b</i>	dist <i>v</i>	wait <i>b</i>	wait <i>v</i>	cost <i>b</i>	cost <i>v</i>	∑ cost	#sync
C101	5.00	7.20	838.69	1239.67	–	–	577.54	1704.80	2282.34	–
C201	1.20	3.00	338.02	748.19	–	–	373.31	1269.09	1642.40	–
R101	4.60	3.00	775.03	801.23	–	–	346.26	656.85	1003.11	–
R201	2.00	2.00	402.93	630.68	–	–	190.03	522.81	712.84	–
RC101	4.00	3.00	668.52	809.78	–	–	296.98	668.67	965.65	–
RC201	2.00	2.00	395.35	668.93	–	–	181.37	552.75	734.12	–
n100-c1	2.00	8.00	56.57	410.36	–	–	184.22	1028.64	1212.86	–
n100-c2	3.00	8.00	63.91	419.69	–	–	204.62	1042.44	1247.07	–
n100-r1	2.00	8.00	39.39	387.64	–	–	120.27	1039.49	1159.75	–
n100-r2	1.00	8.00	23.07	371.74	–	–	75.53	1013.72	1089.25	–
n100-rc1	2.00	8.00	34.24	414.06	–	–	112.32	1066.40	1178.72	–
n100-rc2	2.00	8.00	46.62	379.11	–	–	147.59	1001.75	1149.35	–
n125-c1	3.00	10.00	76.21	517.70	–	–	251.47	1301.61	1553.08	–
n125-c2	3.00	9.00	73.49	461.07	–	–	229.58	1196.10	1425.68	–
n125-r2	1.20	9.00	31.32	404.16	–	–	93.70	1181.51	1275.21	–
n125-r2	2.00	9.00	46.03	408.37	–	–	155.08	1159.25	1314.33	–
n125-rc1	2.00	10.00	35.53	465.73	–	–	116.12	1257.25	1373.38	–
n125-rc2	1.20	9.80	36.53	469.36	–	–	104.26	1249.34	1353.60	–
Vienna	1.80	6.00	45.81	255.41	–	–	117.24	771.86	889.09	–

Table 12 Detailed results for distribution policy with synchronization between vans and cargo bikes

Instance	# <i>b</i>	# <i>v</i>	dist <i>b</i>	dist <i>v</i>	wait <i>b</i>	wait <i>v</i>	cost <i>b</i>	cost <i>v</i>	\sum cost	#sync
C101	5.00	7.80	839.71	1247.25	8.62	1.02	578.76	1727.53	2306.30	5.40
C201	2.00	3.00	433.11	753.29	26.85	33.28	417.27	1275.87	1693.14	3.40
R101	4.40	3.00	761.14	811.91	14.20	41.51	340.82	667.93	1008.75	5.20
R201	2.00	2.00	407.52	633.43	0.00	0.00	191.63	524.49	716.12	2.00
RC101	4.00	3.00	660.51	824.40	0.00	7.96	294.18	678.46	972.64	4.00
RC201	2.00	2.00	394.39	681.87	0.00	0.00	181.04	560.64	741.68	2.00
n100-c1	2.20	8.00	57.81	433.29	5.41	19.16	188.19	1056.61	1244.80	5.60
n100-c2	3.00	8.00	73.67	450.86	10.08	14.20	223.27	1079.54	1302.81	6.20
n100-r1	2.00	8.00	41.15	396.05	16.68	5.14	127.97	1049.75	1177.72	4.00
n100-r2	1.00	8.00	23.07	374.22	2.58	2.40	76.31	1017.04	1093.35	2.00
n100-rc1	2.00	9.00	39.22	456.94	8.06	5.03	122.42	1148.35	1270.77	4.00
n100-rc2	2.00	8.00	43.59	418.19	4.78	5.43	144.36	1043.37	1187.72	4.00
n125-c1	3.20	10.20	84.32	556.32	14.50	28.83	269.11	1376.47	1645.58	9.60
n125-c2	3.00	9.00	74.01	480.42	21.99	8.47	238.78	1220.59	1459.37	7.00
n125-r2	1.20	9.20	31.81	410.91	0.97	2.59	93.08	1195.09	1288.18	2.00
n125-r2	2.00	9.00	47.04	420.83	16.86	7.96	162.30	1175.22	1337.52	4.20
n125-rc1	2.00	10.00	43.38	478.69	0.00	4.15	128.20	1271.82	1400.02	3.00
n125-rc2	1.20	9.80	35.45	477.94	5.60	11.52	101.88	1259.24	1361.12	3.20
Vienna	2.00	6.00	54.80	263.18	7.61	4.17	126.86	779.69	906.56	5.20

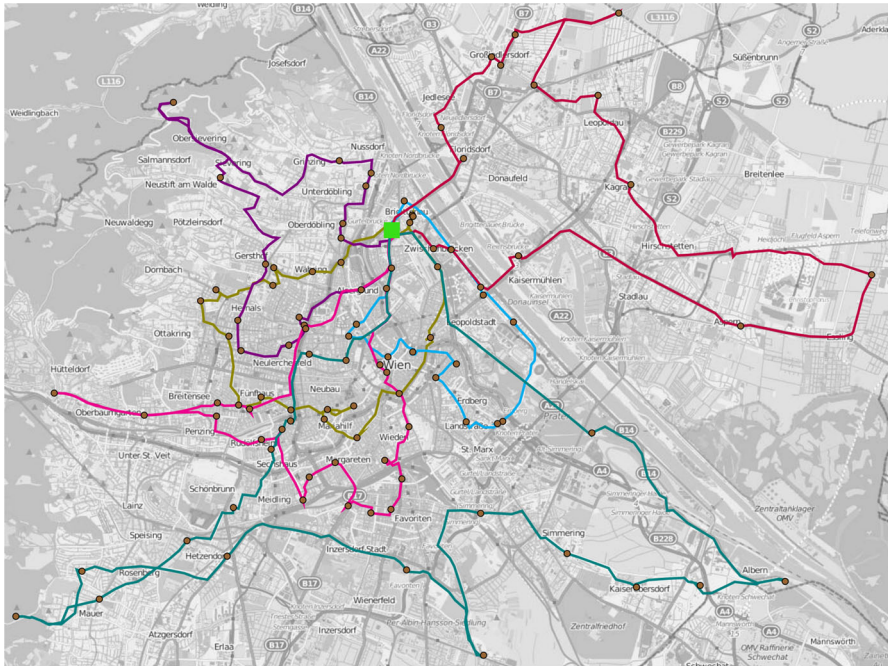


Fig. 7 Sample solution for the Vienna instance, distribution policy: vans only; background map ©OpenStreetMap contributors—www.openstreetmap.org/copyright

and no additional costs in the city center, whereas Fig. 8 displays a sample solution where six vans (solid lines) and two cargo bikes (dashed lines) are used and temporal synchronization takes place at satellites (triangles). Here we penalize van trips crossing the city center as mentioned before.

6 Conclusion

In this paper we present an innovative city distribution scheme for a two-echelon routing problem with temporal and spatial synchronization between vans and cargo bikes. Van trips in the city center should be avoided to mitigate the impact of traffic congestion and noise nuisance. Therefore, cargo bikes perform the last mile distribution in the city center.

We present a GRASP with PR as solution method. Our algorithm provides a first and fast support for decision makers to evaluate when it is—even economically—preferable to use green modes of transport in a city distribution network. Especially the low computational costs of solving this complex problem have to be mentioned, as shown by running our algorithm on a Virtual Machine. We test different possibilities for the inclusion of PR in GRASP. Computational results show that GRASP with integrated PR outperforms GRASP alone in all instances. Using PR also as a post-optimization phase leads to slightly better results in nearly all cases, though the computational time increases significantly.

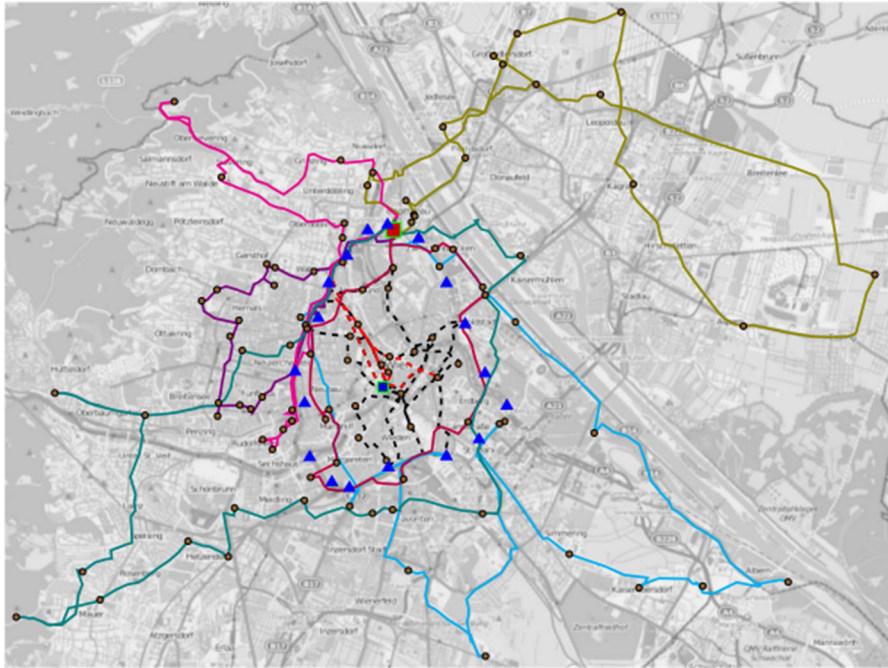


Fig. 8 Sample solution for the Vienna instance, distribution policy: vans (*solid lines*) and cargo bikes (*dashed lines*) with temporal synchronization at satellites (*triangles*); background map ©OpenStreetMap contributors—www.openstreetmap.org/copyright

A comparison of the three distribution policies illustrates that for some instances costs can be saved by the combined usage of cargo bikes and vans instead of vans alone. In all cases emissions can be reduced through the substitution of vans by cargo bikes. In all cases the policy which includes synchronization is more costly than the one without synchronization, when storage costs are not taken into account. Planners can evaluate the trade-off between the additional operational costs of synchronization and the storage costs of satellite facilities.

Nevertheless, certain technical requirements for cargo bikes such as ensuring the required temperature range in the cargo may have to be tackled to use our distribution scheme in a real-world application.

Future research will include the evaluation of CO_2 -emissions as well as uncertainties in travel times and dynamic requests.

Acknowledgements Open access funding provided by Vienna University of Economics and Business (WU). This work is funded by the Austrian Research Promotion Agency as part of the Joint Programming Initiative Urban Europe (FFG Project No. 839739). We gratefully acknowledge this financial support. Furthermore, we thank Markus Straub from AIT for his help with OpenStreetMap and calculating the travel times for our Vienna instance, and we gratefully acknowledge the taxi data provided by Taxi 31300.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

- Baldacci R, Mingozzi A, Roberti R, Calvo RW (2013) An exact algorithm for the two-echelon capacitated vehicle routing problem. *Oper Res* 61(2):298–314
- Beasley J (1983) Route first-cluster second methods for vehicle routing. *Omega* 11(4):403–408
- Brown E, Dury S, Holdsworth M (2009) Motivations of consumers that use local, organic fruit and vegetable box schemes in central england and southern france. *Appetite* 53(2):183–188
- Burke EK, Kendall G (2005) *Search methodologies*. Springer, New York
- Cattaruzza D, Absi N, Feillet D, González-Feliu J (2015) Vehicle routing problems for city logistics. *EURO J Transp Logist* 1–29
- Crainic T, Errico F, Rei W, Ricciardi N (2011a) Modeling demand uncertainty in two-tiered city logistics planning. In: Technical report CIRRELT-2012-65, Université de Montréal, Canada
- Crainic T, Mancini S, Perboli G, Tadei R (2011b) Multi-start heuristics for the two-echelon vehicle routing problem. In: Merz P, Hao J-K (eds), *Evolutionary computation in combinatorial optimization: 11th European conference, EvoCOP 2011, Torino, Italy, April 27–29, 2011, Proceedings, volume 6622 of Lecture Notes in Computer Science*, pages 179–190
- Crainic TG, Gajpal Y, Gendreau M (2012) Multi-zone multi-trip vehicle routing problem with time windows. *CIRRELT-2012-36*
- Crainic TG, Ricciardi N, Storch G (2009) Models for evaluating and planning city logistics systems. *Trans Sci* 43(4):432–454
- Cuda R, Guastaroba G, Speranza M (2015) A survey on two-echelon routing problems. *Comput Oper Res* 55:185–199
- Drexl M (2012) Synchronization in vehicle routing—a survey of VRPs with multiple synchronization constraints. *Transp Sci* 46(3):297–316
- Energieregion (2009) *Mobilitäts- und Marketingkonzept für den Pedelec Einsatz in der Energieregion Weiz-Gleisdorf*
- Feo TA, Resende MG (1989) A probabilistic heuristic for a computationally difficult set covering problem. *Oper Res Lett* 8(2):67–71
- Grangier P, Gendreau M, Lehuédé F, Rousseau L-M (2014) An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization. Technical report, Technical Report 2014-33, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal
- Graser A, Dragaschnig M, Ponweiser W, Koller H, Marcinek M-S, Widhalm P (2012) Fcd in the real world—system capabilities and applications. In: 19th ITS world congress, Vienna, 22/26 Oct 2012
- Hemmelmayer VC, Cordeau J-F, Crainic TG (2012) An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Comput Oper Res* 39(12):3215–3228
- Jepsen M, Spoorendonk S, Ropke S (2013) A branch-and-cut algorithm for the symmetric two-echelon capacitated vehicle routing problem. *Transp Sci* 47(1):23–37
- Juan AA, Pascual I, Guimarans D, Barrios B (2014) Combining biased randomization with iterated local search for solving the multidepot vehicle routing problem. *Int Trans Oper Res* 00:1–21 (online)
- Nguyen PK, Crainic TG, Toulouse M (2013) A tabu search for time-dependent multi-zone multi-trip vehicle routing problem with time windows. *Eur J Oper Res* 231(1):43–56
- Nguyen V-P, Prins C, Prodhon C (2012) Solving the two-echelon location routing problem by a grasp reinforced by a learning process and path relinking. *Eur J Oper Res* 216(1):113–126
- Österreichische Apothekerkammer (2014) *Apotheke in zahlen* 2014
- Perboli G, Tadei R, Vigo D (2011) The two-echelon capacitated vehicle routing problem: models and math-based heuristics. *Transp Sci* 45(3):364–380
- Resende MG, Ribeiro CC (2005) Grasp with path-relinking: Recent advances and applications. In: Ibaraki T, Nonobe K, Yagiura M (eds), *Metaheuristics: progress as real problem solvers*, vol 32 of operations research/computer science interfaces series. Springer, pp 29–63
- Santos FA, Mateus GR, da Cunha AS (2014) A branch-and-cut-and-price algorithm for the two-echelon capacitated vehicle routing problem. *Transp Sci* 49(2):355–368
- World Health Organization—Centre for Health Development (2013) *Annual report 2013*