Solution of Real-World Postman Problems

Stefan Irnich [a,*] Michael Drexl [a]

[a]*Deutsche Post Endowed Chair of Optimization of Distribution Networks, RWTH Aachen University, Templergraben 64, D-52056 Aachen, Germany.*

Abstract

The paper presents the results of a study performed by the Deutsche Post Endowed Chair of Optimization of Distribution Networks in collaboration with Deutsche Post World Net with the aim of improving the planning of letter mail delivery. Modelling and solution methods for real-world postman problems are presented which extend one of the most general postman problems studied in the literature, the windy rural postman problem, with regard to several aspects. The discussed extensions include turn and street crossing restrictions, cluster constraints, the option to have alternative service modes (including 'zigzag deliveries'), and the use of public transport to reach the postal district. The solution method is based on a transformation to the asymmetric TSP and uses non-standard neighbourhood search techniques. Extensive computational experiments show that the solution method clearly and consistently outperforms standard TSP heuristics on real-world instances.

Key words: postman problems, arc routing, transformation into TSP

1 Introduction

This paper describes the results of a study performed by the Deutsche Post Endowed Chair of Optimization of Distribution Networks in collaboration with Deutsche Post World Net with the aim of developing new models and solution algorithms to improve the planning of letter mail delivery.

The letter mail distribution system of Deutsche Post World Net in Germany is a multi-stage, multi-modal network comprising about 110,000 pickup points (letter boxes), 12,000 post offices, 82 regional mail centres, 3,300 delivery depots, and, on the 'last mile', 39 million households and commercial clients expecting more than 20 billion letters per year to be delivered within 24 hours after posting. To perform the delivery, there are 72,000 postmen serving 54,000 districts six days a week on

* Corresponding author.
Email addresses: sirnich@or.rwth-aachen.de (Stefan Irnich), michael@or.rwth-aachen.de (Michael Drexl).

foot, by bicycle or by car. This last stage of letter mail transport incurs several billion Euros of total annual cost. It is clear, therefore, that even a small relative cost reduction amounts to tens of millions of Euros in absolute annual savings. Not surprisingly, Deutsche Post World Net is constantly seeking to optimise the routing of its postmen.

The contribution of the paper is twofold. First, it introduces a model of a very general uncapacitated arc-routing problem capturing most of the practically relevant side constraints encountered by Deutsche Post World Net. In particular, turn and street crossing restrictions, cluster constraints, the option of alternative service modes (including 'zigzag deliveries'), and the use of public transport to reach the postal district are considered. Second, it presents heuristic algorithms for the solution of this model, based on a transformation to the asymmetric travelling salesman problem (ATSP), along with extensive computational results showing the practical applicability of the algorithms.

The rest of the paper is structured as follows: In the next section, the new model is presented and it is shown how the above-mentioned options and additional side constraints can be incorporated. After that, the solution approach is presented: In Section 3, a tailored transformation of the developed arc-routing model to the ATSP is given; in Section 4, it is shown how the resulting ATSP is solved by means of construction and non-standard neighbourhood search improvement procedures. Section 5 presents and analyses the computational experiments demonstrating the superiority of the new algorithmic approach over standard heuristics for the TSP. At the end, Section 6 points to possible interesting extensions of the problem and gives final conclusions.

2 An Extended WRPP Model

There are many models and algorithms for the solution of postman problems (e.g., Eiselt *et al.*, 1995; Dror, 2000). One of the most general models for the uncapacitated case is the windy rural postman problem (WRPP). The WRPP is defined on an undirected graph where some, but not necessarily all, edges require some kind of service and must therefore be traversed at least once. All edges of the graph may be traversed without performing a service an arbitrary number of times in either direction. Traversing an edge incurs a certain cost, and the task is to create a tour for the postman of minimal total cost, covering all required edges. The cost of traversing an edge in one direction may differ from traversing it in the other direction (with and against the wind). Although there are polynomially solvable special cases, the WRPP is, in general, $\mathcal{NP}$-hard (Papadimitriou, 1976). However, the problem is of great significance, as it generalizes many other arc-routing problems. The case where all edges must be traversed, the windy Chinese postman problem (WPP), has received considerable attention in the literature (cf. Minieka, 1979; Guan, 1984; Win, 1987), and there are already some papers dealing with the 'rural' version of this problem as described in the preceding paragraph (Corberán *et al.*, 2005a,b). The paper by Irnich (2005) considers a generalization of the WRPP, the *windy rural postman problem with zigzag service* (WRPPZ). In this problem, the streets (or rather street segments) can be divided into four classes: First, street segments with houses on one side of the street only. These segments must be serviced exactly once. Second, street segments with houses on both sides, where the sides must be serviced separately and where, consequently, the segments must be traversed at least twice. Third, street segments with houses on both sides with the option of servicing both sides simultaneously by a single 'zigzag' traversal (passing through the street segment once) or of servicing the two sides separately by two traversals. Fourth, so-called non-required street segments may be used to get from one point to another. All street segments may be traversed without servicing arbitrarily often in both directions. A traversal without servicing is referred to as *deadheading*. Again, different directions of traversal of a segment may incur different costs. Moreover, also the different traversal modes (one-side service, zigzag service, deadheading) may incur different costs. Formally, the WRPPZ is defined on an undirected graph $G = (V, E)$ with vertex set V and edge set E. Without loss of generality and for notational convenience, it is assumed that G is simple, i.e., that G contains neither parallel edges nor loops. The edge set is partitioned into four subsets, $E = E^0 \cup E^1 \cup E^2 \cup E^3$, where E^0 contains the non-required edges, E^1 and E^2 contain the edges requiring single and double service respectively (but where zigzag service is not allowed), and E^3 contains the edges with zigzag service option. Up to eight different costs c^k_{ij}, c^k_{ji}, $k \in \{0, 1, 2, 3\}$, are associated with an edge $\{i, j\} \in E$. $k = 0$ stands for deadheading, $k = 1$ for servicing the first side, $k = 2$ for servicing the opposite side of the street segment, and $k = 3$ for zigzag service, while ij and ji describe the direction of traversal. Only one direction of traversal may be allowed for an edge for some or all traversal modes. This is represented by setting the corresponding cost coefficient to infinity. Such an edge is called an *arc*. Obviously, for the edges $e \in E \setminus E^3$, only some of the costs are relevant, i.e., edges $e \in E^2$ have six, edges $e \in E^1$ have four, and edges $e \in E^0$ have two different costs. Irnich (2005) gives an integer programming formulation for the WRPPZ.

In this paper, an extended version of the WRPPZ, called EXT-WRPP, is presented. No integer programming formulation for EXT-WRPP is given here; rather, a graph-theoretic approach is taken. This is due to the fact that the study was intended to develop a procedure able to consistently compute good solutions for real-world instances in short time. The size of the instances is such that one cannot expect to achieve this via an exact algorithm, and for the solution approach pursued here, it is not necessary to devise an IP formulation.

In addition to the WRPPZ, EXT-WRPP considers the following aspects:

- Street segment sides: For delivery by foot or by bicycle, it may be necessary to distinguish between a traversal on the left hand side (lhs) or right hand side (rhs) of a street segment. In connection with turns and turn penalties, the consideration of both street sides offers the possibility to favour tours that do not cross main roads too often in order to make the tour faster and safer for the postman. Hence, the deadheading costs are re-defined to reflect traversals from i to j and from j to i, either on the lhs or on rhs of the street: $c_{ij}^{0,lhs}$, $c_{ij}^{0,rhs}$, $c_{ji}^{0,lhs}$, $c_{ji}^{0,rhs}$. W.l.o.g., for edges $e \in E^2 \cup E^3$, the lhs always corresponds to service 1, and the rhs corresponds to service 2.

- Turn restrictions: There is a considerable amount of literature on arc-routing problems with turns, starting with the paper by Caldwell (1961). Other important contributions are the ones by Benavent and Soler (1999); Clossey *et al.* (2001); Corberán *et al.* (2002). Some authors consider the turns issue in their solution algorithms, others use transformations to get equivalent graphs without turns. In this paper, the latter approach is taken. In a simple graph, a turn is unequivocally described by means of three vertices and an associated turn cost/penalty. Forbidden turns are represented by setting the respective penalty to infinity. Hence, when there is no distinction between traversal on the left or the right hand side, the set of possible turns in G is defined as $\mathcal{T} := \{(i,j,k) : \{i,j\},\{j,k\} \in E\}$. The turn cost/penalty is denoted by p_{ijk}. In the case of a detailed model, the street segment side information must also be taken into account. Here, we define $\mathcal{T} := \{(i,s,j,s',k) : \{i,j\},\{j,k\} \in E; s,s' \in \{lhs,rhs\}\}$. The turn cost is then denoted by $p_{ijk}^{s,s'}$. It is assumed that crossing a street segment $\{i,j\}$ where zigzag service is allowed incurs no (measurable) cost. This means that the cost/penalty for the turns out of and into this street segment $\{i,j\}$ are always identical, i.e., $p_{ijk}^{lhs,s} = p_{ijk}^{rhs,s}$ and $p_{kij}^{s,lhs} = p_{kij}^{s,rhs}$ holds for all $s \in \{lhs,rhs\}$ and for all edges $\{i,j\} \in E^3$.
- Clusters: Due to the grouping of street segments into intermediate organizational units (for early morning sorting activities and gathering statistical data), there are clusters of street segments which must be serviced consecutively. More precisely, this means that for two street segments $e, e' \in E$ belonging to the same cluster, if e is serviced before e', there must not be any street segment e'' not belonging to this cluster that is serviced after e and before e'. Deadheading through a street is always allowed. This issue is the arc-routing equivalent to the clustered travelling salesman problem for node-routing problems (see Chisman, 1975). Formally, there is a set of $n_{\mathcal{H}}$ clusters $\mathcal{H} := \{1,\ldots,n_{\mathcal{H}}\}$. The numbers $h_{ij}^{lhs} \in \mathcal{H}$ for $\{i,j\} \in E \setminus E^0$ and $h_{ij}^{rhs} \in \mathcal{H}$ for $\{i,j\} \in E^2 \cup E^3$ denote the cluster index of the service. Note that zigzag service is only possible if both sides of the street segment belong to the same cluster, i.e., we assume $h_{ij}^{lhs} = h_{ij}^{rhs}$ to hold for all $\{i,j\} \in E^3$. The delivery depot (post office) is represented by a single edge $\{i_d, j_d\} \in E^1$. This edge forms a separate cluster.
- Public transport: Sometimes it is not necessary for the postman to deadhead through non-required edges in order to reach his district in the morning or to return to the post office in the afternoon. Districts served on foot or by bicycle can also be reached and left via public transport (buses, tramways, underground). The existing lines are represented in the street distance graph by additional edges connecting end points of otherwise non-adjacent street segments $e \in E^0$. More details are given in Section 3.2.4.
- 'Knock-off after last delivery': A rather new approach in mail delivery operations

is the concept of 'open tours', i.e., the postman ends his workday immediately after the last delivery and returns to the post office only on the next day.

A *postman tour* can be modelled as a triple $P = (C, m, s)$, where $C = (i_0, i_1, \dots, i_n)$ is a walk in G, $m = (m_1, \dots, m_n) \in \{0, 1, 2, 3\}^n$ indicates the traversal modes of the edges in C, and $s = (s_1, \dots, s_n) \in \{lhs, rhs\}^n$ indicates the street segment sides being traversed. A postman tour $P = (C, m, s)$ is feasible if and only if

(P1) (Closed walk) $i_0 = i_n$.
(P2) (Turn feasible) $(i_{p-1}, s_p, i_p, s_{p+1}, i_{p+1}) \in \mathcal{T}$ for all $p \in \{1, \dots, n\}$ (with $i_{n+1} := i_1$ and $s_{n+1} := s_1$).
(P3) (Service covering)

- For all $\{i, j\} \in E^1$, there exists a $p \in \{1, \dots, n\}$ such that $\{i_{p-1}, i_p\} = \{i, j\}$, and $m_p = 1$.
- For all $\{i, j\} \in E^2$, there exists a $p \in \{1, \dots, n\}$ such that $\{i_{p-1}, i_p\} = \{i, j\}$, and $m_p = 1$.
- For all $\{i, j\} \in E^2$, there exists a $p \in \{1, \dots, n\}$ such that $\{i_{p-1}, i_p\} = \{i, j\}$, and $m_p = 2$.
- For all $\{i, j\} \in E^3$, there exist $p_1, p_2 \in \{1, \dots, n\}$ such that $\{i_{p_1-1}, i_{p_1}\} = \{i_{p_2-1}, i_{p_2}\} = \{i, j\}$ and either ($p_1 = p_2$ and $m_{p_1} = 3$) or ($p_1 \neq p_2$, $m_{p_1} = 1$, and $m_{p_2} = 2$).

(P4) (Cluster feasible) For all $1 \leq p < r \leq n$, the conditions $m_p \neq 0$, $m_r \neq 0$, and $h^{s_p}_{i_{p-1}, i_p} = h^{s_r}_{i_{r-1}, i_r} =: h$ imply $m_q = 0$ (deadheading) or $h^{s_q}_{i_{q-1}, i_q} = h$ (same cluster) for all $p < q < r$.
(P5) (Compatibility between m_p and s_p) For all $1 \leq p \leq n$, $m_p = 1 \Leftrightarrow s_p = lhs$ and $m_p = 2 \Leftrightarrow s_p = rhs$.

Note that (P5) does not impose any restriction about the side of the street if the street is deadheaded or serviced in zigzag mode.

It is assumed that *each* solution fulfilling (P1–P5) is feasible with respect to the postman's carrying capacity and working time regulations. Let the set of all postman tours P fulfilling (P1–P5) be $\mathcal{P}^{feas}$.

The *cost* of a postman tour $P = (C, m, s)$ with $C = (i_0, i_1, \dots, i_n)$ is

$$c(P) = c(C, m, s) := \sum_{j=1}^{n} \left(c^{m_j}_{i_{j-1} i_j} + p^{s_j, s_{j+1}}_{i_{j-1}, i_j, i_{j+1}} \right), \tag{1}$$

where, again, $i_{n+1} := i_1$ and $s_{n+1} := s_1$. With these definitions, EXT-WRPP can be stated formally as

$$\text{(EXT-WRPP)} \quad z^\star = \min_{P \in \mathcal{P}^{feas}} c(P). \tag{2}$$

3 Problem Transformation

The model for EXT-WRPP presented in the preceding section is a descriptive one, but it is well-defined in a graph-theoretical sense. To transform the problem into an

ATSP, three different graphs are considered: (a) the undirected graph $G = (V, E)$ for the representation of the problem instance; (b) a directed graph $D' = (V', A', c')$ to model and compute shortest paths w.r.t. distance and turn penalties; (c) a complete digraph $\bar{D} = (\bar{V}, \bar{A}, \bar{c})$ on which the resulting ATSP is defined.

The basic idea of the transformation is that each service is represented by two anti-parallel arcs, each of which indicates the direction of travel when performing the service. Services of different street segments are connected by directed arcs representing deadheadings from the end point of the first to the start point of the second service. The associated costs are computed as shortest path distances in the digraph D' and take into account costs for deadheading and turns. Services of the same street segments can be connected to either model consecutive but separate services or—using a similar techniques as in (Irnich, 2005)—to model zigzagging.

3.1 Street Distance Digraph

The street distance digraph $D' = (V', A', c')$ models the physical street network including costs for deadheadings and turns. Any path in D' is an alternating path where arcs either represent deadheadings (on a specific side of the street and in a given direction) or feasible turns. For each street segment $e = \{i, j\} \in E$ there are two directions, ij and ji, and two sides, lhs and rhs. Hence, four arcs with eight distinct nodes i^{side}_{dir} and j^{side}_{dir} with $dir \in \{ij, ji\}$ and $side \in \{lhs, rhs\}$ are necessary to represent the possible deadheadings. The four nodes i^{side}_{dir}, $dir \in \{ij, ji\}, side \in \{lhs, rhs\}$ are associated with the endpoint i, i.e., one end of the street segment, while the four nodes j^{side}_{dir} represent the other end. Thus, the deadheadings of street segment $\{i, j\} \in E$ are given by the four arcs $(i^{lhs}_{ij}, j^{lhs}_{ij})$, $(i^{rhs}_{ij}, j^{rhs}_{ij})$, $(j^{lhs}_{ji}, i^{lhs}_{ji})$, $(j^{rhs}_{ji}, i^{rhs}_{ji})$ with costs $c^{0,lhs}_{ij}, c^{0,rhs}_{ij}, c^{0,lhs}_{ji}$, and $c^{0,rhs}_{ji}$, respectively.

Let $t = (i, s, j, s', k) \in \mathcal{T}$ be an arbitrary feasible turn. Since $\{i, j\}, \{j, k\} \in E$, the nodes j^s_{ij} and $j^{s'}_{jk}$ are already given by the above definition. The turn t is uniquely represented in D' by the arc $(j^s_{ij}, j^{s'}_{jk}) \in A'$ with cost $p^{s,s'}_{ijk}$. Finally, we denote by d'_{kl} the shortest path distance between an arbitrary pair of nodes $k, l \in V'$ in the street distance digraph.

3.2 ATSP Digraph

The main tool to solve the postman problem defined in Section 2 is its transformation into an ATSP. Each feasible postman tour $P = (C, m, s)$ corresponds to a Hamiltonian cycle in the complete ATSP digraph $\bar{D} = (\bar{V}, \bar{A}, \bar{c})$ with the same cost (except for multiples of a constant M, see below). Furthermore, a cost-minimal Hamiltonian cycle $\bar{C}$ in $\bar{D}$ encodes an optimal postman tour. The decoding, i.e., the transformation of the ATSP solution into the corresponding postman tour, can be done in time proportional to the length of the tour. For the sake of brevity, any Hamiltonian cycle in $\bar{D}$ is called (ATSP) tour.

Note that the proposed transformation does not provide a one-to-one correspondence between feasible postman tours and ATSP tours. While feasible postman tours have associated Hamiltonian cycles in $\bar{D}$, most ATSP tours do not imply feasible postman tours. The transformation, therefore, has to utilise a constant M, a sufficiently big number, such that ATSP tours not representing a feasible postman tour are more costly than feasible ones.

The transformation of the problem instance into an ATSP is explained in three steps: First, we describe *internal arcs* connecting some of the pairs of nodes belonging to the same edge $e \in E$. The intention of an internal arc is to model a service that is performed in a particular orientation. Consequently, costs of internal arcs mainly consist of costs defined by the costs c^k_{ij}, c^k_{ji}, $k \in \{1, 2, 3\}$. Second, *external arcs* connect nodes belonging to different edges of E or nodes of the same edge but for services on opposite sides of the street connected by deadheading. Their cost is, therefore, defined as the length of a path in the street distance graph. Third, we describe the transformation of a (feasible) ATSP solution into the associated postman tour.

Note that the following transformations are similar to those presented in (Irnich, 2005) for the WRPPZ. However, turn restrictions and cluster constraints impose several modifications regarding structure and cost of arcs. These modifications are pointed out in the following.

3.2.1 Internal Arcs of the ATSP Digraph

First, for each edge $e = \{i, j\} \in E^1$, there are two nodes $k := i^{lhs}_e$ and $\ell := j^{lhs}_e$. The two nodes are connected by anti-parallel arcs $(k, \ell) = (i^{lhs}_e, j^{lhs}_e)$ and $(\ell, k) = (j^{lhs}_e, i^{lhs}_e)$ with costs $c^1_{ij} - M$ and $c^1_{ji} - M$, respectively. The intention of adding $-M$ to both arcs is that a sufficiently large number M ensures that an optimal ATSP tour contains either (i^{lhs}_e, j^{lhs}_e) or its anti-parallel counterpart (j^{lhs}_e, i^{lhs}_e). This first case and the following two cases are depicted in Figure 1.

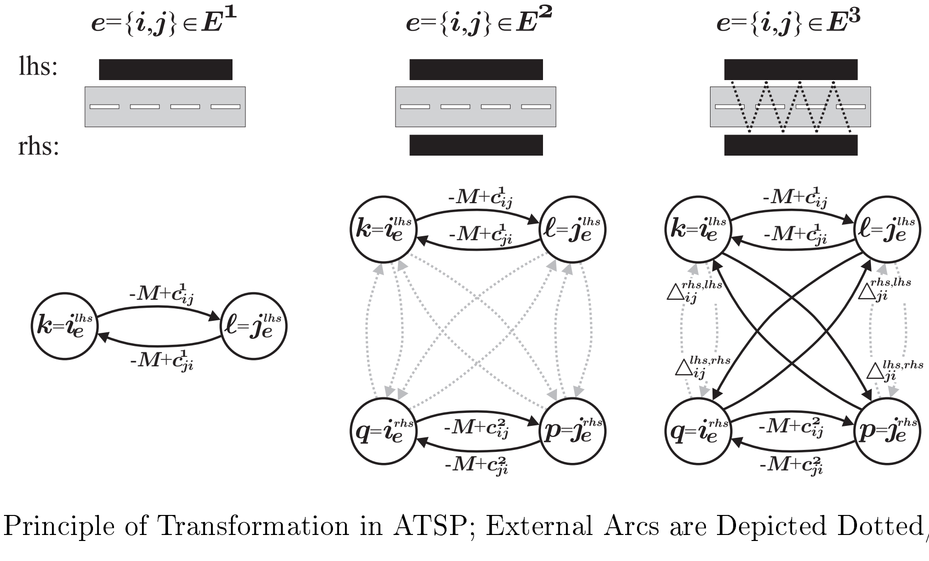


Fig. 1. Principle of Transformation in ATSP; External Arcs are Depicted Dotted/Greyed

Second, for each edge $e = \{i, j\} \in E^2$, there are two associated services and, hence, four nodes $k := i^{lhs}_e$, $\ell := j^{lhs}_e$, $p := i^{rhs}_e$, and $q := j^{rhs}_e$. The four nodes are connected by the following internal arcs. For the left hand side of the street, there are two anti-parallel arcs $(k,\ell)=(i_e^{lhs},j_e^{lhs})$ and $(\ell,k)=(j_e^{lhs},i_e^{lhs})$ with costs $c^1_{ij}-M$ and $c^1_{ji}-M$, respectively. Similarly, for the right hand side, the arcs $(p,q)=(i_e^{rhs},j_e^{rhs})$ and $(q,p)=(j_e^{lhs},i_e^{lhs})$ have costs $c^2_{ij}-M$ and $c^2_{ji}-M$. In the following, these arcs are called *service arcs*. The eight remaining connections between these four nodes are external arcs and will be described in the next subsection.

Third, for each edge $e=\{i,j\}\in E^3$, there are again four nodes $k:=i_e^{lhs}$, $\ell:=j_e^{lhs}$, $p:=i_e^{rhs}$, and $q:=j_e^{rhs}$. In addition to the four arcs for edges $e\in E^2$, we add four other internal arcs: The arc (ℓ,p) models two alternative options at the same time. If (ℓ,p) is used in an optimal solution, the two arcs (k,ℓ) and (p,q) for the two associated services are also used, directly before and after arc (ℓ,p). On the one hand, this is possible when the associated services are connected by deadheading. On the other hand, the option of zigzagging through the street segment $e=\{i,j\}$ is also represented by the sub-path (k,ℓ,p,q). Hence, the cost of (ℓ,p) is defined as

$$\Delta_{ij}^{lhs,rhs}:=\min\{c^3_{ij}-c^1_{ij}-c^2_{ij},d'_{j_{ij}^{lhs},i_{ij}^{rhs}}\}. \tag{3}$$

Note that the first term of the definition is defined such that the cost of zigzagging is $(c^3_{ij}-c^1_{ij}-c^2_{ij})+(c^1_{ij}-M)+(c^2_{ij}-M)=c^3_{ij}-2M$. If the minimum in (3) is given by the second term, the cost of the sub-path (k,ℓ,p,q) is exactly the cost $c^1_{ij}+c^2_{ij}$ of the two services plus deadheading costs minus $2M$. This definition is consistent with the two options which had to be modelled. Similarly, the connections (q,k), (k,q), and (p,ℓ) are assigned costs $\Delta_{ij}^{rhs,lhs}$, $\Delta_{ji}^{lhs,rhs}$, and $\Delta_{ji}^{rhs,lhs}$, respectively.

3.2.2 External Arcs of the ATSP Digraph

All remaining arcs (k,k') of $\bar{D}$ have costs given by shortest path distances d' in the street distance graph D' and additional penalties of M if the two sides of the street belong to different clusters. Each node, k and k', has a unique associated service. Thus, let $\{i,j\}\in E$ be the street segment, $s\in\{lhs,rhs\}$ be the side of the street, and h^s_{ij} be the cluster of the service associated with k. Moreover, let ij be the direction of travel such that k lies on *end*point j of the street segment. Similarly, let $\{i',j'\}=\{j',i'\}$, $s'\in\{lhs,rhs\}$, $i'j'$, and $h^{s'}_{i'j'}$ be the associated values for node k', but defined such that k' lies at the *start*point i' of the associated service. Then, the arc $(k,k')\in\bar{A}$ has cost

$$\bar{d}_{kk'}=d'_{j^s_{ij},i'^{s'}_{i'j'}}+\begin{cases}M,\text{ if } h^s_{ij}\neq h^{s'}_{i'j'}\\ 0,\text{ otherwise}\end{cases}.$$

3.2.3 Validity of the Transformation

In order to show the validity of the transformation, we have to make sure that an optimal ATSP tour in $\bar{D}$ corresponds to a feasible, cost-minimal postman tour in G. In the following, we first show how (optimal) ATSP tours can be transformed back into postman tours $P=(C,m,s)$. Second, we explain why optimal ATSP tours necessarily create feasible postman tours. Third, the fact that the proposed transformation is cost-preserving for feasible postman tours is easy to see, so that the validity of the whole transformation follows.

The transformation from ATSP tours to postman tours constructs the walk $C = (i_0, i_1, \dots, i_n, i_0)$ from several paths $(j_0, \dots, j_p)$ imposed by the arcs or arc sequences of the ATSP tour. On the one hand, external arcs of $\bar{D}$ uniquely correspond to deadheadings, i.e., node sequences $(j_0, \dots, j_p)$ with edge traversal modes $(m_1, \dots, m_p) = (0, \dots, 0)$, and specific street sides $(s_1, \dots, s_m)$. On the other hand, we have to distinguish several cases for the internal arcs. Note first that there are $S := |E^1| + 2(|E^2| + |E^3|)$ services to cover, which are represented by $2S$ pairwise anti-parallel service arcs in $\bar{D}$. Thus, any ATSP tour can collect a profit of $-S \cdot M$ at best. Moreover, there are $n_{\mathcal{H}}$ clusters, and penalties of $+M$ are put on all inter-cluster external arcs. For sufficiently large M, therefore, any optimal ATSP tour uses exactly S service arcs and exactly $n_{\mathcal{H}}$ inter-cluster external arcs (this implies property (P4), and, with the following arguments, also (P3)).

The use of exactly S service arcs also implies that arcs labelled with Δ (these arcs exist only for edges $\{i, j\} \in E^3$ and are named $(k, q), (q, k), (p, \ell), (\ell, p)$) can only be traversed in the sequences $(k, \ell, p, q), (q, p, \ell, k), (\ell, k, q, p)$, or (p, q, k, ℓ). Depending on whether the minimum in (3) is given by the first or second term, the corresponding node sequences, services and traversal modes have to be computed differently. As mentioned above, for the sequence (k, ℓ, p, q) and $\bar{d}_{\ell,p} = c^3_{ij} - c^1_{ij} - c^2_{ij}$ (minimum given by first term) the cost of the sequence (k, ℓ, p, q) is $c^3_{ij} - 2M$. The corresponding sequences in G are therefore (i, j), $m = (3)$, and s arbitrarily defined (recall the assumption stated in last sentence of paragraph on turn restrictions in Section 2). Similar arguments apply to the three other sequences. If the minimum in (3) results from the second term, the two associated services are performed consecutively but not in zigzag mode. For the sequence (k, ℓ, p, q) in $\bar{D}$, the resulting part of the postman tour is given by $(i, j, h_2, \dots, h_{p-2}, i, j)$, $m = (1, 0, \dots, 0, 2)$, and $s = (lhs, s_2, \dots, s_{p-1}, rhs)$. The sub-path $(j, h_2, \dots, h_{p-2}, i)$ corresponds to a cost-minimal deadheading from the end-point of the first service of $\{i, j\}$ (performed in direction ij, located on the lhs) to the start-point of the second service of $\{i, j\}$ (performed in direction ij, located on the rhs). Again, similar arguments apply to the three other sequences.

The transformation of service arcs corresponding to edges $\{i, j\} \in E^1 \cup E^2$ must be performed separately for each pair of anti-parallel service arcs. The same holds for service arcs corresponding to edges $\{i, j\} \in E^3$ when the service arc is not incident to an internal arc labelled with Δ. In all these cases, the service is on an edge $\{i, j\}$, and, depending on the associated direction, the part of the postman tour is either (i, j) or (j, i) with service mode and traversal either $m = (1)$ and $s = (lhs)$, or $m = (2)$ and $s = (rhs)$.

The above transformation ensures that consecutive arcs (k, k') and $(k', k'') \in \bar{A}$ impose node sequences that fit together, i.e., sequences where the last node implied by (k, k') is also the first node implied by (k', k''). Summing up, these arguments show that the resulting postman tour is closed (P1) and—because of the use of turn-feasible paths from the street distance network—fulfills property (P2). The compatibility (P5) is also guaranteed in all of the above cases. It is straightforward to check that all steps of the transformation are cost-preserving. Hence, cost-minimal postman tours correspond to optimal ATSP tours.

3.2.4 Integration of Public Transport and Single Park-and-Loop Operations

There are two practical requirements that can also be handled with the above transformation. First, postmen might use public transport to get from the delivery depot to their postal district and vice versa. These additional options for deadheading can easily be integrated into the original graph G by using additional edges $e \in E^0$ and lead to additional arcs in the street distance network between those points where a public transport connection exists. The resulting enlarged street distance network, denoted by D'_0, can be used instead of D'. Mostly, the use of public transport is only permitted between the delivery depot and the first/last point of the district. In this case, only the deadheadings between the 'service' $\{i_d, j_d\} \in E^1$ and other services should be computed in D'_0 instead of D'. The 'knock-off after last delivery' concept fits nicely into this model, since connections back to i_d can be set to cost zero.

Furthermore, different means of transport might be used for reaching or leaving the district and the delivery operations. For instance, the postman might use a car to get to a remote district and use a trolley (on foot) to service households and corporate clients. This case can be solved *iteratively* with the proposed transformation. One simply has to replace the delivery depot by potential parking places p, solve the EXT-WRPP with 'depot' p, and add the cost for the deadheading between the delivery depot and the parking place. The cost-minimal solution of these is the solution to this single park-and-loop operations problem. The case that a car is stopped multiple times to perform several loops in different parts of the district has been considered in the work of (Bodin and Levy, 2000). However, these more complicated routing problems are beyond the scope of this paper.

3.3 Transformation into an STSP

ATSP instances can be transformed into STSP instances using the transformation of Jonker and Volgenant (1983). Each ATSP node i is duplicated into two STSP nodes i_{in} and i_{out} which have to be visited consecutively, i.e., they are connected with (negative) cost $-M$. ATSP arcs (i, j) with cost $\bar{d}_{ij}$ are transformed into edges $\{i_{out}, j_{in}\}$ with the same cost and all remaining connections in the new graph are infeasible, i.e., edges $\{i_{out}, j_{out}\}$ and $\{i_{in}, j_{in}\}$ have cost $+M$. We use this transformation mainly in order to compute lower bounds using branch-and-cut techniques for STSP. Moreover, we will compare standard edge-exchange procedures for the STSP to the transformed postman problem and compare these results with our specialised ATSP and arc-routing improvement procedures presented in the next section.

4 Solution Methods for ATSP

The book chapter (Johnson *et al.*, 2002) covers the standard solution methodology with constructive and improvement heuristics for the ATSP. For the sake of brevity, we restrict ourselves to explain only the new aspects of our solution method.

4.1 Construction Heuristics

ATSP construction heuristics rely on the well-known *nearest neighbour* and *greedy* (also called *multiple fragment*) principles or are based on solving assignment problems as relaxations of the ATSP, such as variants of the *patching heuristic* that were studied by Glover *et al.* (2001). We implemented several of these heuristics and found out that the nearest neighbour heuristic—despite its simplicity—is the best one to compute ATSP tours corresponding to *feasible* postman tours. The 'pathological' instances, where nearest neighbour solutions imposed infeasible postman tours, had cluster constraints, delivery by car, and a large fraction of one-way streets. Thus, the nearest neighbour heuristic was used for the ATSP tour construction.

4.2 Improvement Heuristics

We use a variant of an *iterated* or *chained local search* heuristic which consists of the following components. First, edge-exchange neighbourhoods that can be searched in $\mathcal{O}(n^2)$ time, such as 2-opt, Or-opt, and string-exchange, are combined such that a local optimum w.r.t. all these neighbourhoods is computed. Hansen and Mladenović (2001) refer to this principle as variable neighbourhood descent (VND). Note that these neighbourhoods together are able to eliminate all infeasibilities resulting from using Ms in the transformations. Local optima always correspond to a feasible postman tour. Second, an extension of the restricted dynamic programming neighbourhood of Balas and Simonetti (2001) is then applied to the result. This neighbourhood can change a large number of edges at the same time. It is linear in the tour length n, but exponential in a parameter $k \geq 2$, and thus is a very large-scale neighbourhood. It was shown by Balas and Simonetti (2001) that it provides a reasonable complement to the classical ATSP edge-exchange procedures, which only modify a few edges. If an improvement is found both neighbourhood search procedures are repeated. Finally, a local minimum w.r.t. all neighbourhoods results. Unbiased random double-bridge moves (called *kick* moves) perturb the current tour and provide new *iterated* start solutions for the above described local search descent heuristic. The following sections provide more details about the basic components of the improvement procedures.

4.2.1 Edge-Exchange Procedures

A 2-opt move inverts a sub-string of the current tour, an Or-opt move relocates a string, and a string-exchange move exchanges two strings (see, e.g., Funke *et al.*, 2005). As long as the relocated and exchanged strings are of limited length, sequential search methods (Irnich *et al.*, 2006) based on the *gain criterion* (Lin and

Kernighan, 1973) allow the acceleration of local search such that best improving moves can be identified faster than by scanning all $\Theta(n^2)$ possible moves. We use this technique for the Or-opt neighbourhood. Instead, the dynamic programming procedure of Glover (1996) is used for determining a best double-bridge move in $\mathcal{O}(n^2)$ time (i.e., a string-exchange move without limitations on the string length resulting in a $\Theta(n^4)$-sized neighbourhood).

In order to avoid that the edge-exchanges directly undo the double bridge kick moves, the VND procedure first uses the 2-opt and Or-opt neighbourhoods and applies double-bridge moves only to local optima w.r.t. 2-opt and Or-opt.

4.2.2 Very Large-Scale Neighbourhood Search

Very large-scale neighbourhood search (VLSNS) is a variant of local search in which the search for a best neighbour solution of a large neighbourhood is transformed into another optimization problem that can be solved in (pseudo-)polynomial time. The term 'large' refers to neighbourhoods that cannot be inspected by enumeration of neighbour solutions—one by one—in an 'acceptable' time. A family of large-scale neighbourhoods, one for each integer $k \geq 2$, has been proposed and analysed by Balas and Simonetti (2001). Given an ATSP tour $x = (x_0, x_1, \ldots, x_n, x_0)$ the neighbourhood $\mathcal{N}^k_{BS}(x)$ consists of all routes $x' = (x_{\pi(0)}, x_{\pi(1)}, \ldots, x_{\pi(n)}, x_{\pi(0)})$ where the permutation π of $\{0, 1, \ldots, n\}$ fulfills the following conditions: For any two indices $i, j \in \{0, 1, \ldots, n\}$ with $i + k \leq j$ the inequality $\pi(i) \leq \pi(j)$ holds. The meaning of this definition is that if node x_i precedes node x_j by k or more positions then x_i must also precede x_j in the neighbour solution.

The search for a best neighbour solution $x' \in \mathcal{N}^k_{BS}(x)$ is performed by solving a shortest-path problem in an auxiliary graph $G^* = G^*_k$ (we will leave out the index k in the following). The auxiliary graph G^* is well-structured; it consists of $n+2$ stages for a tour x of length $n+1$. Each (typical) stage i consists of $(k+1)2^{k-2}$ nodes V_i. Only consecutive stages i and $i+1$ are linked, i.e., there are $k(k+1)2^{k-2}$ arcs joining V_i with V_{i+1}. Stage 0 contains the start node o and stage $n+2$ the sink node d. Every o-d-path in G^* is in one-to-one correspondence to a neighbour solution x'. The correspondence is implied by the fact that each node j refers to a specific position in the tour x. One of the amazing properties of G^* is that the structure of all consecutive stages does not depend on i. The induced subgraphs $G^*(V_i \cup V_{i+1})$ are all identical, they neither depend on i nor on n, but only on k. This allows computing $G^*(V_i \cup V_{i+1})$ beforehand in order to construct the entire auxiliary graph G^* quickly.

Figure 2 shows an example of G^* for $k = 3$ and a tour of length $n + 1 = 6$. Every node $j \in V_i$ refers to position $i + \alpha(j)$, i.e., every row in Figure 2 defines an offset $\alpha(j)$ relative to the current position i. Those nodes that refer to positions $i + \alpha(j) < 0$ or $i + \alpha(j) > n + 1$ are nodes that cannot be reached on any o-d-path. Stages i with unreachable nodes are called *untypical stages*; corresponding nodes and arcs are shown dotted in Figure 2. In order to ensure that any o-d-path in G^* has a cost identical to the cost of the implied neighbour solution x', one has

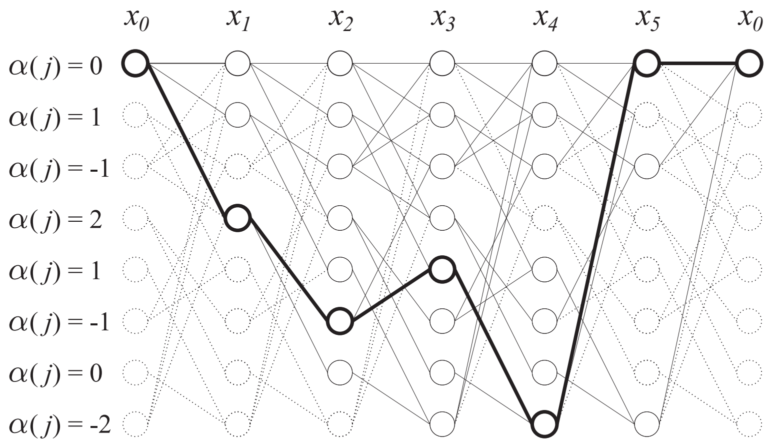


Fig. 2. Auxiliary Graph G^* for $k = 3$ and $n + 2 = 7$ Stages

to label arc $(j, j') \in V_i \times V_{i+1}$ with cost $c_{x_{i+\alpha(j)}, x_{i+1+\alpha(j')}}$. For instance, the first bold arc in Figure 2 is labelled with cost $c_{x_{0+0}, x_{1+2}} = c_{x_0, x_3}$, the second has cost $c_{x_{1+2}, x_{2+(-1)}} = c_{x_3, x_1}$ etc. This is perfectly in accordance with the fact that the bold path represents the neighbour tour $x' = (x_0, x_3, x_1, x_4, x_2, x_5, x_0)$.

It is obvious that we can apply VLSNS using the neighbourhood $\mathcal{N}^k_{BS}$ to ATSP tours resulting from the transformation described in Section 3. However, we further adapted the idea to arc-routing problems which are transformed into node-routing problems by replacing a service arc with two nodes. The main drawback of a direct application to transformed postman tours is that individual nodes but not arcs are permuted. If one wants to exchange arcs over k (arc) positions, one has to move nodes over $2k$ positions. However, service arcs can be reversed and it is, therefore, not sufficient to simply rearrange the sequence of the arcs. Here, we consider inverted and non-inverted arcs as atoms for the neighbourhood search procedure.

More precisely, we propose to construct a modified neighbourhood $\tilde{\mathcal{N}}^k_{BS}$ by duplicating the nodes of G^* such that $j \in V_i$ stands for a non-inverted service arc and $j' \in \tilde{V}_i$ refers to its inverted counterpart. For a fixed number k, the new auxiliary graph $\tilde{G}^*$ has twice the number of nodes per (typical) stage as the original auxiliary graph G^*. It has four times the number of arcs between stages. Given a postman tour $x = (o, a_1, a_2, \ldots, a_n, d)$ with service arcs $a_1, a_2, \ldots, a_p$, start-node o and end-node d, we build $\tilde{G}^*$ with $n + 2$ stages and $(k + 1)2^{k-1}$ nodes $V_i \cup \tilde{V}_i$ for each stage. Each arc $(j, j') \in V_i \times V_{i+1}$ of the original auxiliary graph has three fellow arcs $(j, \tilde{j}') \in V_i \times \tilde{V}_{i+1}$, $(\tilde{j}, j') \in \tilde{V}_i \times V_{i+1}$, and $(\tilde{j}, \tilde{j}') \in \tilde{V}_i \times \tilde{V}_{i+1}$. Hence, $\tilde{G}^*$ has $(n + 1)k(k + 1)2^k$ arcs. An example of a transformed postman tour $(o, a_1, a_2, a_3, a_4, d)$ with $n = 4$ service arcs is depicted in Figure 3. From the values $\alpha(j)$, we see that the bold path permutes the arcs a_1, a_2, a_3, a_4 such that the new ordering is a_2, a_1, a_4, a_3. At the same time, the arcs a_1 and a_4 are inverted, because the stages 2 and 3, which are referring to the arcs $a_{2+(-1)} = a_1$ and $a_{3+1} = a_4$, visit nodes of $\tilde{V}_2$ and $\tilde{V}_3$, respectively. Given that all arcs a_j have tail and head nodes p_j and q_j, the old tour x is $(o, p_1, q_1, p_2, q_2, p_3, q_3, p_4, q_4, d)$, and the

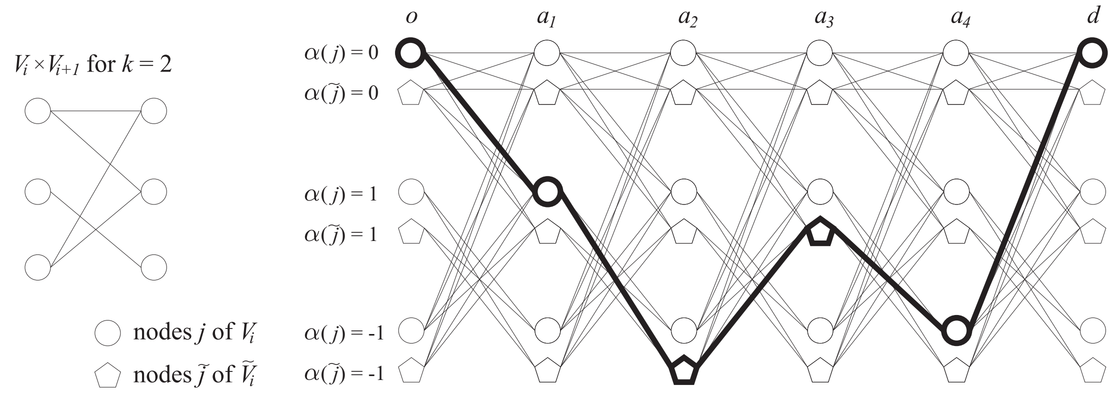


Fig. 3. Construction of the Auxiliary Graph $\tilde{G}^*$ from G^* for $k = 2$

new tour is $x' = (o, p_2, q_2, q_1, p_1, q_4, p_4, p_3, q_3, d)$. The advantage of the modification proposed in this section is that for a repositioning of arcs over less than k positions, we only need to use $\tilde{\mathcal{N}}^k_{BS}$ on graph $\tilde{G}^*$ instead of $\mathcal{N}^{2k}_{BS}$ on graph G^* for the ATSP. Since the number of nodes and arcs grow exponentially with k, the factor 2 is crucial. The example shows that all arcs are moved one position forward or backward (which requires $k = 2$ for $\tilde{G}^*$), while some nodes are moved up to three positions (which requires $k = 4$ for G^*).

5 Computational Results

The following computational analysis is based on real-world instances provided by our project partner, Deutsche Post World Net. All postal districts considered here were taken from the urban part of the city of Aachen (Germany), which is a mid-sized city with about 250,000 inhabitants. Most of the postal districts allow delivery on foot or by bicycle, only a smaller fraction requires delivery by car. EXT-WRPP instances were created from these 56 postal districts by altering (1) the location of the associated delivery depot; (2) the means of transport used for servicing the district, depending on whether streets/roads of the district allow delivery on foot, by bicycle, or by car; (3) weights for the importance of turn penalties w.r.t. service and traversal costs; and by deciding (4) whether or not cluster constraints have to be respected (operational vs. tactical planning). The objective was always to create a time-minimal postman tour, i.e., traversal and service costs were measured in time units. A validated model for estimating the time of servicing a street segment by zigzagging was not available. Certainly, these service times depend on the distance between consecutive entrances of houses on each side of the street, the width of the street and pavement, the flow of traffic through the street, the means of transport etc. A simulation model in combination with comprehensive collection of statistical data could possibly help to identify the most important influencing variables, but the effort for such an empirical study was beyond the scope of the actual project. For the sake of simplicity, costs for zigzag deliveries were assumed to be identical to the costs of separate services for delivery on foot and by bicycle. For delivery by car, zigzagging is only possible on very sparsely populated road segments; we assumed that the costs for zigzagging are 2.5 times higher than the cost for separate deliveries. Similarly, we used estimates for the temporal effort to perform turns by classifying the set of all allowed turns into left-turns, straight crossings, right-turns, and u-turns, taking also the means of transport into account. Concluding, 1236 test instances resulted from this setup, 672 with delivery on foot, 432 by bicycle, and 132 by car.

The majority of the edges defining the EXT-WRPP graph $G = (V, E)$ are edges $e \in E^0$, because the entire street network of the city can in principle be used for deadheading from the end of one service to the beginning of the next service. The number $|E|$ of edges ranges from 4018 to 4499 (with avg. 4373), while the number of edges of a particular type is between 6 and 71 (avg. 21.6) for $|E^1|$, between 0 and 47 (avg. 14.2) for $|E^2|$, between 0 and 39 (avg. 9.4) for $|E^3|$, and between 19 and 100 (avg. 45.2) for required edges $R = E \setminus E^0$. The resulting ATSP instances had between 62 and 258 nodes with an average of 137.6 nodes.

All problem-specific algorithms were coded in C++, compiled in release mode with MS-Visual C++ .NET 2003 version 7.1; all runs were performed on a standard PC (Intel x86 family 15 model 2) with 2.0 GHz, 1GB main memory, on MS-Win 2000. The free STSP solvers Concorde and Linkern (version Dec 19, 2003) were run under the Linux emulator Cygwin (version 1.5.10-3).

5.1 Exact and Heuristic Algorithms for STSP

With the final transformation of the EXT-WRPP instances into STSP instances described in Section 3.3 we can apply well-known and freely available STSP solvers to our problem. The Concorde branch-and-cut solver of Applegate *et al.* (1999) is an exact algorithm which provides optimal solutions to our easy-to-solve problem instances. For the others it gives lower bounds, so that heuristic approaches can be analysed w.r.t. solution quality.

A first remarkable observation of the branch-and-cut code is that its running time is highly unpredictable and can scatter between less than 1 second and more than 20 hours for our small-sized STSPs (on average 275 nodes). There is nearly no correlation between the size of the STSP instance and the running time. As a consequence, we skipped trying to compute exact solutions for all 1236 instances, but Concorde was used to create a lower bound by solving the root node of the branch-and-cut tree only. The lower bound results from solving the 2-matching relaxation of the STSP and adding different types of cutting planes to it. Note that the Concorde implementation is not able to handle instances with negative costs on the edges (as resulting from our transformations). Hence, multiples of M were added to all rows/columns of the STSP distance matrix so that the resulting instance had no negative costs. However, we observed that the lower bounds sometimes differ if several calls of Concorde are performed on the same instance. We suspect that Concorde uses some random components to heuristically identify violated cuts. For our statistical analysis this means that parts of the results are not fully and consistently reproducible.

Besides the exact solver, Applegate *et al.* (1999) freely distribute the heuristic solver

Linkern for STSPs, which is based on a chained Lin-Kernighan edge-exchange procedure (Lin and Kernighan, 1973; Martin *et al.*, 1992). Applying Linkern to our STSP instances provides upper bounds ub and enables us to compute an optimality gap $(ub - lb)/lb$. The results can be summarised by the following key facts: 40 out of 1236 instances were solved to proven optimality (Linkern and Concorde gave identical bounds). Contrary, 7 times Linkern was not able to compute a feasible postman tour, since the STSP tour still had some dispensable edges with cost greater than M. The optimality gap provided by Linkern and Concorde (without branching) was 5.6% on average with a maximum of 35.0% (over all feasible tours found by Linkern). Unexpectedly, Linkern was 2 times (out of 1236 instances) able to beat all of our algorithms (see next section). In those cases, the Linkern result was 2.98% and 1.92% better than the best solution computed by our algorithms. Nevertheless, the gap between Linkern and our best found solution was on average 4.2% worse than our solutions and 29.8% worse in the maximum.

Concluding, the freely available STSP solution methods do not turn out satisfactory in solving the STSP instances resulting from the EXT-WRPP transformations. The large gaps and resulting long running times of the branch-and-cut algorithm are an indication that these small-sized instances are very hard to solve from a practical point of view. We suppose that this difficulty results from the high degeneracy of the instances, where multiples of M dominate the objective. More tailored solution approaches are needed, such as the combined edge-exchange and VLSNS approach empirically analysed in the following section.

5.2 Classical Edge-Exchanges and the Balas & Simonetti Neighbourhood

There are two main parameters that influence the quality and speed of the proposed iterated VND approach combining classical edge-exchange and VLSNS techniques: First, kicks for the perturbation of local optima can be applied as long as one wants to continue the search. One can in general expect that the overall time increases linearly with the number of kicks and that better solutions are found if more kicks are used. The tradeoff between solution quality and running time is recorded in our setup in the following way: We keep track of the best solution computed so far and report this best solution together with the running time at six points in time: before the first kick (in the following denoted by $kick = 0$) and after $100, 200, 500, 1000$, and 2000 kicks. Second, the Balas & Simonetti neighbourhood and its modification $\tilde{\mathcal{N}}_{BS}^{k}$ proposed in Section 4.2.2 have a parameter k for the distance over which nodes/arcs can be re-positioned. Although we expect that larger values of k allow finding better solutions, results for an actual instance can show a different behaviour: For $k_1 < k_2$, the best solution finally found using $\tilde{\mathcal{N}}_{BS}^{k_1}$ may be better than the one found with $\tilde{\mathcal{N}}_{BS}^{k_2}$. Note that the combination of the classical edge-exchange and the VLSNS procedures can create different local optima depending on the interplay of these neighbourhoods. We therefore compare the different combinations of the parameters $kick$ and k statistically over a large set of test instances.

First, we analyse the running time of the different algorithms. Figure 4 depicts the average running times over all 1236 instances. Each of the curves shows the

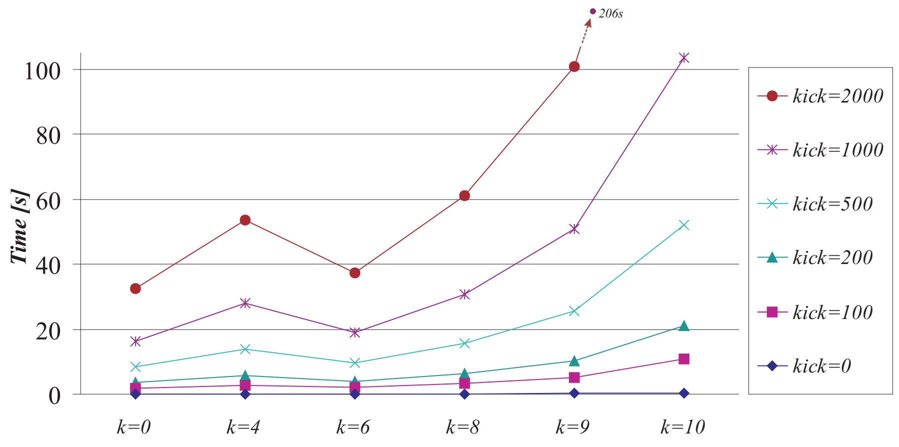


Fig. 4. Average Running Times w.r.t. Parameters k and $kick$

running times for a fixed number of kicks and for increasing values of k with $k = 0$ for a setup without the modified Balas & Simonetti neighbourhood and $\tilde{\mathcal{N}}^k_{BS}$ with $k \in \{4, 6, 8, 9, 10\}$. Average running times range from less than 1 second for $kick = 0$, to an average of 206 seconds for the most expensive combination with $kick = 2000$ and $k = 10$. Amazingly, the overall running times show a non-increasing behaviour w.r.t. k, which one might not have expected: Solving the ATSPs with the smaller neighbourhood $\tilde{\mathcal{N}}^4_{BS}$ took longer than with the larger neighbourhood $\tilde{\mathcal{N}}^6_{BS}$. The reason for this unexpected behaviour is that for $k = 6$ less iterations with the classical edge-exchange neighbourhoods were necessary to reach a mutual local optimum. The larger running times for a single search step in $\tilde{\mathcal{N}}^6_{BS}$ compared to $\tilde{\mathcal{N}}^4_{BS}$ were overcompensated by the better quality of the computed neighbour solutions. Similar reasons can explain why the algorithms resulting from $\tilde{\mathcal{N}}^4_{BS}$ and $\tilde{\mathcal{N}}^8_{BS}$ consume nearly the same amount of time (for an identical number of kicks).

Next, we analyse the solution quality of the different algorithms. Since we do not know optimal solutions for all instances, there exist two possible ways to measure solution quality. A pessimistic viewpoint is that the proposed heuristics fail to find good solutions, so that the lower bound lb computed by CONCORDE is the only reasonable value to compare solutions with. Let ub_A be the best solution found by algorithm A. Then $gap_A = (ub_A - lb)/lb$ is the relative deviation from the CONCORDE lower bound lb of algorithm A. Note that all these values are defined for a fixed instance of the problem. Since we are interested in statistical evaluations we only report average (=mean) and maximum values over the 1236 instances of our test set. The optimistic viewpoint is that at least one of our algorithms gets (very close to) the optimal solution. Defining $ub_{best} = \min_A ub_A$, the relative deviation (in percent) of algorithm A from the best solution found is $gap_{A,best} = (ub_A - ub_{best})/ub_{best}$. (In the following, we will leave out the index A.) Figures 5 and 6 display the average gaps gap and gap_{best} over all 1236 instances.

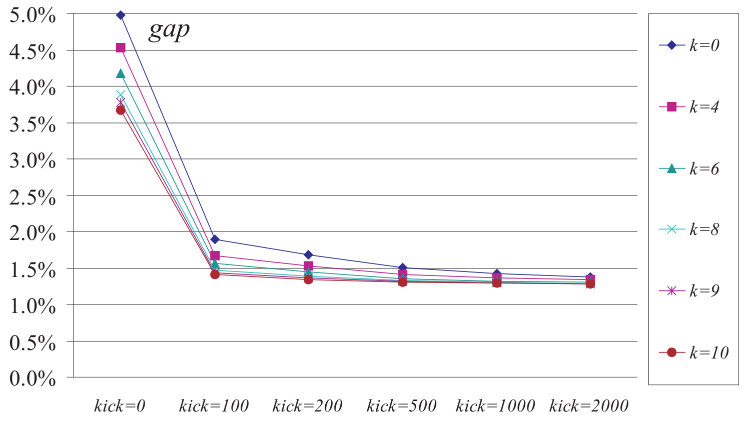


Fig. 5. The Average Values of *gap* Computed by Different Algorithms for Parameters *kick* and k

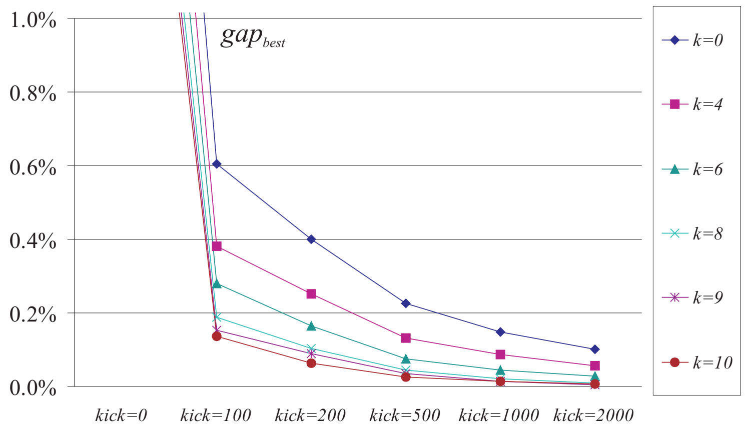


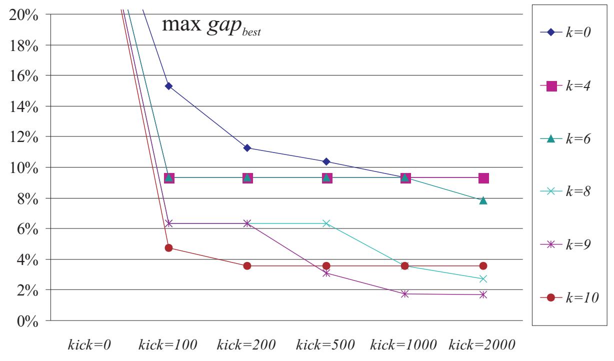
Fig. 6. The Average Values of gap_{best} Computed by Different Algorithms for Parameters *kick* and k

In contrast to Figure 4, each curve in the Figures 5 and 6 shows the improvement w.r.t. parameter *kick* but for a fixed value of k. Note the different scales of the two diagrams. The deviation *gap* from the lower bounds provided by CONCORDE is on average less than 2% when at least 100 kicks are performed. The deviation goes down to less than 1.5% for $kicks \geq 500$. It seems that neither more kicks nor greater values of k help to reduce the average deviation *gap*. At this point, it was not clear to us whether the large gaps are caused by weak lower bounds provided by CONCORDE or weak upper bounds computed by our algorithms. In order to analyse the bounds in more detail, we considered instances where the deviation *gap* was maximal. The maximal gap *gap* for the comparison with the CONCORDE lower bound is large, i.e., $gap = 28.4\%$. By applying the full branch-and-cut algorithm to the worst-case instance (with 312 nodes in the STSP) we found out that the gap of was caused by both the weak lower bound computed by CONCORDE and the weak solution computed by LINKERN: The output was $lb = 11{,}798$, $ub_{best} = 15{,}144$. The full branch-and-cut running time of CONCORDE was 47 seconds on 19 branch-

and-bound nodes with an optimal solution $z^\star = 12{,}243$. All of our algorithms with $k \geq 4$ delivered $gap_{best} \leq 1.9\%$ after 100 kicks and found this optimal solution after 200 kicks. Similar results were observed for other instances with a large gap. We therefore think that gap_{best} provides a more realistic picture of the solution quality.

This average deviation gap_{best} from the best solution found is depicted in Figure 6. It is very small and allows the following interpretations: If classical edge-exchange neighbourhoods and the modified Balas & Simonetti neighbourhoods are applied together in the VND approach, the resulting algorithm consistently finds better solutions with the same number of kicks. On the one hand, even the smallest value of $k = 4$ permits reducing the average gap_{best} by a factor of approx. 1.7 compared to algorithms that do not use the modified Balas & Simonetti neighbourhood. For $k = 6$ the reduction is by about factor 3, for $k \geq 8$ the reduction is by factors of between 6 and 9. On the other hand, doubling the number of kick moves (or going from 200 to 500 kicks) also consistently improves the quality of the solutions: The reduction of the average deviation gap_{best} is here by factors of between 1.4 and 2.9.

Another pessimistic viewpoint is the consideration of the maximum deviation gap_{best} over all 1236 instances, referred to as $\max gap_{best}$. These values are shown in the diagram in Figure 7 for all combinations of $kick$ and k. All initial VND so-



$k =$	$kick =$ 100	200	500	1000	2000
0	575	438	283	214	145
4	482	349	208	136	84
6	384	270	144	86	42
8	296	196	103	61	22
9	271	182	84	48	16
10	247	139	70	39	19

Fig. 7. The Maximum Values of gap_{best} and Number of Instances where ub_{best} was Missed

lutions (i.e., $kick = 0$) have $\max gap_{best} = 30.4\%$, independent of the parameter k. After 100 kicks, for algorithms not using the Balas & Simonetti neighbourhood, this value reduces to about 15%, while algorithms with the VLSNS approach have values of less than 9%. For $k = 4$, more kicks did not improve the result. Contrary, for $k \geq 6$, additional kicks help to reduce $\max gap_{best}$ continuously. For $k = 6$ the gap is in the maximum smaller than 8% and for $k \geq 8$ less than 4% after 1000 kicks. Moreover, the small table in Figure 7 shows the number of times a specific algorithm did not find the best known solution. For instance, numbers smaller than 62 mean that in more than 95% of the cases (out of 1236 instances) the best known solution was computed. All this is a clear indication that the modified Balas & Simonetti neighbourhood with $k \geq 6$ in combination with an appropriate number of kicks makes the algorithms behave robustly so that they consistently produce high-quality solutions.

Finally, we present the results concerning the tradeoff between solution quality and the time spent on searching. Each point $(gap_{best}, time)$ in Figure 8 shows the averages over all 1236 instances for a specific combination of $kick$ and k. Additionally, the area of dominated points is shaded in grey. For the sake of visibility, points with

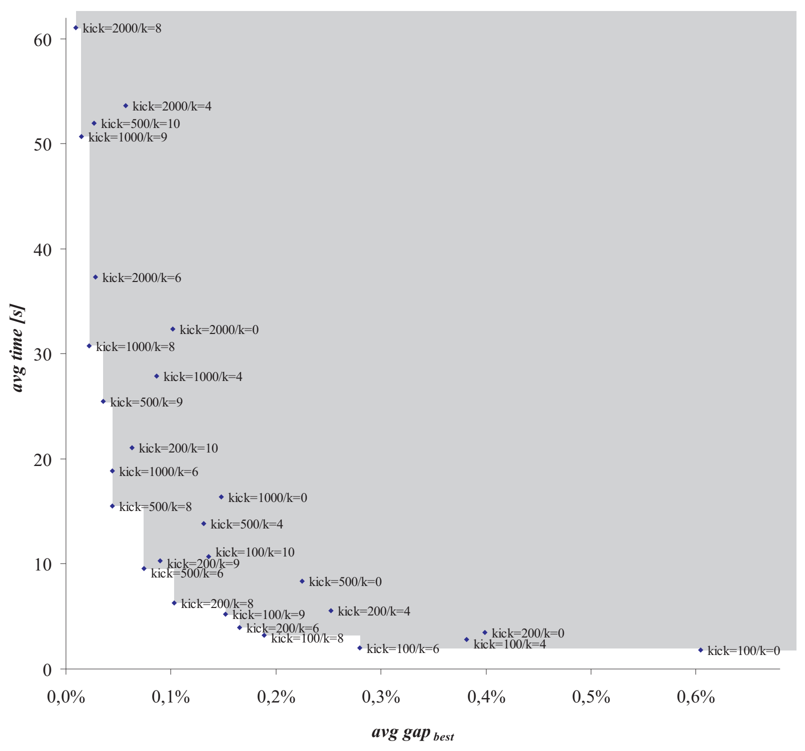


Fig. 8. Tradeoff between Quality and Computation Time

computation times larger than 65 seconds are not shown. These are the combinations $(kick, k) = (2000, 9), (1000, 10)$, and $(2000, 10)$ with values $(0.008\%, 206.5s)$, $(0.013\%, 103.5s)$, and $(0.005\%, 100.9s)$. In essence, the best solutions were obtained with these computationally most costly setups. Moreover, larger average gaps of $gap_{best} > 0.7\%$ are also not displayed in Figure 8. These result from the initial solutions of the VND component, i.e., from algorithms with $kick = 0$. The corresponding running times are all very small, but the solution quality in the average case and—in particular—in the worst case is unacceptably bad (as was already discussed in the previous paragraphs). We interpret the remaining results in the following way: Only combinations with $k = 6$, $k = 8$, and less often, with $k = 9$ are Pareto-optimal w.r.t. running time and average solution quality. Except for high-quality and minimum-time combinations (not depicted), combinations with $k = 0$, $k = 4$ and $k = 10$ lead to dominated algorithms: Apparently, large values of k are too time consuming because of the exponential growth in k of the running time for searching $\tilde{\mathcal{N}}^k_{BS}$. Similarly, small values of k lead to fast algorithms, but they lack robustness and quality of solutions. As a general recommendation (for instances resulting from transformed postman problems), the values $k = 6$ and $k = 8$ produce well-performing algorithms and the tradeoff between quality and running time can be controlled by choosing an appropriate number of kicks.

6 Conclusions

This paper has considered the EXT-WRPP, a very general uncapacitated arc-routing problem incorporating several practically relevant constraints and options such as street segment sides, turn penalties, clustered street segments and the possibility of servicing street segments in zigzag mode. The problem is of great practical importance for letter mail delivery. In Germany alone, there are more than 50,000 districts to be serviced every day, each district corresponding to an EXT-WRPP instance. To formally describe the problem, a graph-theoretic model has been presented. For solving this model, the paper has proposed a transformation to a standard node-routing problem without additional constraints, namely, to an asymmetric or symmetric TSP. Computational experiments show that the solution of the resulting TSP instances with standard algorithms does not work satisfactorily. This is most probably due to the highly degenerate cost structure of the instances stemming from the use of big M constants. Therefore, a new heuristic solution method has been devised, based on VND with kicks, using classical edge-exchange and VLSNS steps. By computational experiments Pareto-optimal parameter settings have been identified yielding solutions clearly and consistently outperforming standard TSP heuristics with respect to solution quality and computing time. Moreover, the optimal settings also exhibited a very robust behaviour in the average and worst case in tests on a large set of real-world instances.

Further interesting research directions include the consideration of time windows and the extension to multiple vehicles/carriers. In the latter case, one has to take vehicle and temporal carrier capacities into account. The resulting arc-routing problems require to simultaneously cluster street segments into districts and decide on the routing and the mode of servicing. Particularly relevant for Deutsche Post World Net is the clustering of the service areas according to a given ratio of full time and part time postmen.

We think that this paper constitutes a contribution to more realistic models and effective solution approaches in arc-routing. Certainly, more integrated network-design, location, and routing models for postal applications will be developed in the near future.

Acknowledgement

We would like to thank our project partners from Deutsche Post World Net, in particular Ralf Lösche, Ulrich Plösser, Dr. Uwe Prantz, and Dr. Norbert Will, and also our supervisor, Prof. Hans-Jürgen Sebastian from Deutsche Post Endowed Chair of Optimization of Distribution Networks, for initiating this research project, providing us with data and additional useful information and, last but not least, for the kind collaboration.

References

Applegate, D., Bixby, R., Chvátal, V., and Cook, W. (1999). CONCORDE. Available at http://www.math.princeton.edu/tsp/concorde.html.

Balas, E. and Simonetti, N. (2001). Linear time dynamic-programming algorithms for new classes of restricted TSPs: A computational study. *INFORMS Journal on Computing*, **13**(1), 56–75.

Benavent, E. and Soler, D. (1999). The directed rural postman problem with turn penalties. *Transportation Science*, **33**(4), 408–418.

Bodin, L. and Levy, L. (2000). Scheduling of local delivery carrier routes for the united states postal service. In Dror (2000), chapter 11, pages 419–442.

Caldwell, T. (1961). On finding minimum routes in a network with turn penalties. *Communications of the ACM*, **4**, 107–108.

Chisman, J. (1975). The clustered traveling salesman problem. *Computers & Operations Research*, **2**, 115–119.

Clossey, J., Laporte, G., and Soriano, P. (2001). Solving arc routing problems with turn penalties. *Journal of the Operational Research Society*, **52**, 433–439.

Corberán, A., Martí, R., and Sanchis, J. (2002). A GRASP heuristic for the mixed chinese postman problem. *European Journal of Operational Research*, **142**, 70–80.

Corberán, A., Plana, I., and Sanchis, J. (2005a). A branch & cut algorithm for the windy general routing problem. Technical report, Department of Statistics and OR, University of Valencia, Spain.

Corberán, A., Plana, I., and Sanchis, J. (2005b). The windy general routing polyhedron: A global view of many known arc routing polyhedra. Technical report, Department of Statistics and OR, University of Valencia, Spain.

Dror, M., editor (2000). *Arc Routing: Theory, Solutions and Applications*. Kluwer, Boston.

Eiselt, H., Gendreau, M., and Laporte, G. (1995). Arc routing problems, Part I: The chinese postman problem. *Operations Research*, **43**(2), 231–242.

Funke, B., Grünert, T., and Irnich, S. (2005). Local search for vehicle routing and scheduling problems: Review and conceptual integration. *Journal of Heuristics*, **11**(4), 267–306.

Glover, F., Gutin, G., Yeo, A., and Zverovich, A. (2001). Construction heuristics for the asymmetric TSP. *European Journal of Operational Research*, **129**, 555–568.

Glover, F. (1996). Finding a best traveling salesman 4-opt move in the same time as a best 2-opt move. *Journal of Heuristics*, **2**, 169–179.

Guan, M. (1984). On the windy postman problem. *Discrete Applied Mathematics*, **9**, 41–46.

Hansen, P. and Mladenović, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, **130**(1), 449–467.

Irnich, S., Funke, B., and Grünert, T. (2006). Sequential search and its application to vehicle-routing problems. *Computers & Operations Research*, **33**(8), 2405–2429.

Irnich, S. (2005). A note on postman problems with zigzag service. *INFOR*, **43**(1), 33–39.

Johnson, D., Gutin, G., McGeoch, L., Yeo, A., Zhang, W., and Zverovitch, A. (2002). Experimental analysis of heuristics for the ATSP. In G. Gutin and A. Punnen, editors, *The Traveling Salesman Problem and Its Variations*, volume 12 of *Combinatorial Optimization*, chapter 10, pages 445–487. Kluwer, Dordrecht.

Jonker, R. and Volgenant, T. (1983). Transforming asymmetric into symmetric traveling salesman problems. *Operations Research Letters*, **2**, 161–163.

Lin, S. and Kernighan, B. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, **21**, 498–516.

Martin, O., Otto, S., and Felten, E. (1992). Large-step Markov chains for the TSP incorporating local search heuristics. *Operations Research Letters*, **11**(4), 219–224.

Minieka, E. (1979). The chinese postman problem for mixed networks. *Management Science*, **25**(7), 643–648.

Papadimitriou, C. (1976). On the complexity of edge traversing. *Journal of the ACM*, **23**, 544–554.

Win, Z. (1987). *Contributions to routing problems*. Ph.D. thesis, Universität Augsburg, Augsburg.