Centro de Investigação Operacional

Local search heuristics for sectoring routing in a household waste collection context

Maria João Cortinhal, Maria Cândida Mourão, Ana Catarina Nunes

CIO – Working Paper 1/2015

Local search heuristics for sectoring routing in a household waste collection context

Maria João Cortinhal[b,c,*], Maria Cândida Mourão[a,c], Ana Catarina Nunes[b,c]

[a] *Instituto Superior de Economia e Gestão, ULisboa, Rua do Quelhas 6, 1200-781 Lisboa, Portugal*
[b] *ISCTE-IUL – Instituto Universitário de Lisboa, Av. das Forças Armadas, 1649-026 Lisboa, Portugal*
[c] *Centro de Investigação Operacional, ULisboa, 1749-016 Lisboa, Portugal*

Abstract

This paper addresses the problem of residential waste collection, as a real life application of a sectoring-arc routing problem (SARP). The aim is to design the service area and the set of planned trips for each vehicle, whilst minimising a stated objective. The sectoring-arc routing problem simultaneously handles both tactical and operational decisions – sectors and the planning of vehicle trips – and thus avoids successive sub-optimisation. In order to obtain solutions that satisfy features required by real life applications, three criteria are taken into account: total routing time, workload time imbalance between sectors, and service connectivity within each sector. We propose a two-phase heuristic which favours solutions in which total routing time and the non-connectivity of the service area of each vehicle are minimised. A hill climbing and a tabu search based heuristic are also derived. By means of a normalised, weighted sum of criteria for evaluating solutions, the local search heuristics were tailored to maintaining the best features of the initial solutions, whilst minimising workload time imbalance. The algorithms are tested in random instances and also in real life based instances. The results show that combining the two-phase heuristic with a local search method is an efficient way of obtaining good quality solutions to implement in practice: namely connected service areas with a balanced workload that result in a negligible increase in total routing time. They also highlight that the proposed function for evaluating solutions during the search phase plays an essential role.

Keywords: Routing, District design, Heuristics, Capacitated arc routing.

1. Introduction

Waste collection systems cover different types of waste, such as residential, commercial, recyclable or skip waste. Depending on the characteristics and location of the waste containers, vehicle trips are tackled via node or arc routing problems.

*Corresponding author.
Email addresses: maria.cortinhal@iscte.pt (Maria João Cortinhal), cmourao@iseg.ulisboa.pt (Maria Cândida Mourão), catarina.nunes@iscte.pt (Ana Catarina Nunes)

This paper addresses a residential waste collection problem, in which waste is collected along the streets by a fixed number of capacitated vehicles. Thus, a capacitated arc routing approach is considered. Moreover, depending on streets size and on traffic rules, some streets can only be serviced in one direction, whereas others demand collection on both sides and in both directions. Large one-way streets need to be represented by multiple segments, and consequently the street network is represented by a mixed multigraph.

This research work was motivated by real life applications. Therefore, the design of vehicle trips is threefold. Firstly, it should optimise total routing time. Secondly, the street network must be partitioned into a fixed number of sub-regions (also called *sectors*, or *service areas*), all with similar workload times, and each sub-region should be as connected as possible. Thirdly, each sector must be serviced by a single capacitated vehicle, which performs one or more trips within a limited workload time due to labour regulations. It should be noted that improving connectivity, whilst at the same time imposing a time limit on sectors, promotes solutions in which service areas are both geographically concentrated and grouped into delimited regions.

The problem under analysis can thus be modelled as a sectoring-arc routing problem (SARP) (Mourão et al., 2009). The SARP combines a sectoring problem (also known as districting, district design, or territory design) with an arc routing problem (ARP). More precisely, in this study a mixed capacitated arc routing problem (MCARP) is tackled, as the street network is a mixed one and vehicle capacity is limited. The capacitated arc routing problem (CARP) is an optimisation problem that was first introduced by Golden and Wong (1981) for undirected networks. As the CARP is a NP-hard problem (Golden and Wong, 1981), so is the MCARP, and thus the SARP is also, as the latter reduces to the former, if only one sector is considered. Therefore, for large sized instances, heuristic methods are the most suitable approach for obtaining high-quality solutions in reasonable computational time.

In this paper we propose a two-step solution method that has been devised to build high-quality solutions that can be put into practice by practitioners. The first step consists of a two-phase heuristic that constructs initial solutions by minimising the total routing time, whilst favouring connectivity within the service areas. The second step is an improvement method, which is designed to provide better balanced sectors, whilst keeping, or even enhancing, whenever possible, the good characteristics of the sectors that were developed in the first step. For this purpose we suggest two local search algorithms, one of which is based on hill climbing, and the other of which is a tabu search. Both local search algorithms make use of a weighted function to evaluate solutions during the search process. To take the planning criteria suggested by the practitioners into account, this function integrates three different measures of quality. Furthermore, as a means of increasing the performance of the local search algorithms, which requires computation times that generally increase with the dimension of the instances, the list of possible moves within the neighbourhood is restricted by means of a time-distance measure, which is similar to the restricted candidate list used in the construction phase of the GRASP metaheuristic (Feo and Resende, 1995).

To the best of the authors' knowledge, the solution methods suggested in this paper have never been addressed in the literature on the SARP. Moreover, this paper contributes to the

literature in this field by providing some insights into the benefits of integrating different measures of quality for evaluating solutions during the search process.

The remainder of this paper is organised as follows: in Section 2 we review the related literature, and in Section 3 the problem is described; the constructive two-phase heuristic and the local search solution methods are detailed in Sections 4 and 5 respectively; and then the computational results are summarised and analysed in Section 6, before the conclusions, which are given in Section 7.

2. Literature review

The construction of trips has been widely addressed in the literature, most of which is devoted to node routing approaches, as may be confirmed in Golden et al. (2008), and in Toth and Vigo (2014). Extensive surveys regarding arc routing can be found in Dror (2000), Wøhlk (2008), Corberán and Prins (2010), and Corberán and Laporte (2014).

The design of sectors has been considered for multiple purposes, such as political districting (Bozkaya et al., 2003; Bação et al., 2005), commercial territory design (Ríos-Mercado and Fernández, 2009; Jarrah and Bard, 2012; Salazar-Aguilar et al., 2012), road maintenance (Muyldermans et al., 2003; Perrier et al., 2008), meter reading (Assis et al., 2014) and also waste collection (Teixeira et al., 2004; Lin and Kao, 2008; Mourão et al., 2009; Constantino et al., 2015).

Partitioning problems for routing purposes usually demand some degree of specificity. Among these, connectivity appears within each sector and sectors balance. As pointed out by Perrier et al. (2008), each sector must be balanced in workload, and should be contiguous in the sense that the subgraph induced by its demand units should be connected.

Sectors balance is a feature which is required to obtain sectors similar in size. In the literature, several balance measures have been considered, such as the ones based on: i) trips duration (Kim et al., 2006; Mourão et al., 2009) or on its estimate (Gonzalez-Ramírez et al., 2011); ii) the length of the links serviced (Perrier et al., 2008); iii) the quantity serviced (Mourgaya and Vanderbeck, 2007; Salazar-Aguilar et al., 2012); iv) a relationship between quantity and travelled distance (Teixeira et al., 2004; Lin and Kao, 2008), or; v) on the number of customers (Salazar-Aguilar et al., 2012). Despite being different, all these measures are meant to evaluate the workload assigned to each sector. Therefore, to some extent, sectors balance increases the chance of a fair distribution of workload amongst the different members of the team.

In the literature, sectors balance is promoted through different ways, namely: i) by constraints that impose upper bounds on cost (Haugland et al., 2007), on the demand (Mourgaya and Vanderbeck, 2007), or on the length of each sector (Perrier et al., 2008); ii) by tailored evaluation functions (Lin and Kao, 2008; Ríos-Mercado and Fernández, 2009; Gonzalez-Ramírez et al., 2011; Salazar-Aguilar et al., 2012; Assis et al., 2014), or; iii) sometimes by the solution method as a whole (Teixeira et al., 2004; Kim et al., 2006; Mourão et al., 2009).

Leaving sectors apart, the balancing requirement is also referenced in the routing literature that discusses the construction of vehicle trips. For instance, both Jozefowiez et al.

(2009) and Oyola and Løkketangen (2014) address the capacitated vehicle routing problem with trip balancing. In this extension of the problem, two minimisation objectives are tackled: the difference between the longest and the shortest trip length (*makespan*), and also the usual total length.

The connectivity within each sector is related to the contiguity of its demand units, and therefore to the possibility of reaching each other within their service area. For this purpose, some authors suggest constructive heuristics that consider specific rules to select and add the demand units to a sector, which additionally favour the concentration of each service area in a geographical sub-region (Muyldermans et al., 2003; Haugland et al., 2007; Mourão et al., 2009), while Constantino et al. (2015) propose exact and heuristics methods based on models that evaluate the node overlapping of different service zones. These characteristics enable both service specialisation and the allocation of responsibilities to work teams.

Other authors consider connectivity explicitly as a constraint, as is the case of the local search based methods of Ríos-Mercado and Fernández (2009) and Lei et al. (2012). On the other hand, the heuristic solution methods proposed by Perrier et al. (2008) consist of solving MILP models in which connectivity is explicitly imposed by linear constraints.

As referred to earlier, the SARP addressed in this paper is a problem which involves simultaneous tactical and operational decisions and, respectively, sectors design and the planning of vehicle trips. The idea is to avoid successive sub-optimisation by embedding operational activities, or their estimates, in strategic or tactical decisions (see e.g. Salhi and Rand, 1989; Simchi-Levi, 1992; Cattrysse et al., 1997; Ghiani and Laporte, 2001). Mainly focussed on strategic and tactical issues, Ghiani et al. (2014) survey waste management systems.

Few studies combine the design of service areas with the definition of trips, as is the case of Teixeira et al. (2004), Kim et al. (2006), or Ramos and Oliveira (2011) for node routing applications, and Mourão et al. (2009), or Constantino et al. (2015) for the arc routing case.

The heuristic solution approach of Teixeira et al. (2004) for a recyclable waste collection case study is a constructive three phase method which aims to minimise operation costs. Firstly, a zone per vehicle is defined, taking the area and the population into account, in order to balance their collection effort. Then, for each sector, the last two phases determine the type of waste to be collected, and a single trip for each day of the month.

Kim et al. (2006) address a case study for a commercial waste collection vehicle routing problem with time windows. The aim is to group stops into clusters, and to then build a single vehicle trip for each cluster, whilst minimising total travel time. The authors propose an extended insertion algorithm and a clustering-based algorithm, both of which are enhanced with a simulated annealing improvement method. The clustering-based algorithm promotes the workload balance of the routes and also improves the proximity among the stops of the same route.

In their recyclable waste collection case study, Ramos and Oliveira (2011) present a constructive heuristic which simultaneously obtains the service area for each depot and also vehicle trips. Based on an estimated collecting time, the main goals are the minimisation of the distances and the balancing of workload at the depots.

Mourão et al. (2009) developed three heuristics for a sectoring arc routing problem.

Sectors are limited in their workload time and the aim is to minimise total routing time. The heuristics promote connectivity, as well as workload time balance among sectors. A best insertion method builds sectors and vehicle trips simultaneously. In the two-phase heuristics, the sectors are built in phase 1 by two alternative procedures, both of which consider workload time estimates. Phase 2 executes a mixed CARP heuristic to obtain the vehicle trips within each sector.

Focused on real life applications, Constantino et al. (2015) developed solution methods for the mixed capacitated arc routing problem with limited overlapping of the vehicle service routes. The proposed exact and heuristic methods are based on MILP models, in which an upper bound is imposed on the number of nodes shared by different routes. This upper bound is previously obtained via MILP models that minimize the number of nodes shared by different vehicle services.

Although the design of vehicle trips is not included, the design of service areas for routing approaches may benefit from considering routing estimates (trip cost or duration). To illustrate this, we refer to Haugland et al. (2007), Jarrah and Bard (2012), and Lei et al. (2012), all of whom study sector design methods for stochastic vehicle routing problems.

3. Problem description

The household waste collection problem studied focused on what can be described as follows; a network represents the street segments (*links*), some of which require a service (waste collection), whilst others do not (no waste collection). Each *required street* segment, which is also named as a *task*, is symbolised: i) by one edge, if it is a narrow two-way street that allows simultaneous collection for each of its two sides (zigzag or parallel collection); ii) by two opposite arcs, if it is a large two-way street, with each side collected separately; iii) by one arc, if it is a one-way street, or; iv) by two parallel arcs, if it refers to a large one-way street that must be serviced twice, once for each side. Each task has a *service time*, which measures the time for collecting its waste. As some required street segments need to be represented by edges, a special data structure is used and each edge $u = (i, j)$ is then replaced by two inverse linked arcs, u and $inv(u) = (j, i)$ (Lacomme et al., 2001). Henceforward, we therefore refer to required edges as arcs.

The *non-required street* segments, or *deadheading links*, are always represented by one arc or two arcs, depending on whether they relate to a one-way or two-way street, respectively. Furthermore, deadheading is allowed, which means that even required streets can be traversed without being serviced. Therefore, each arc has a *deadheading time*, which represents the time required to traverse it.

A homogeneous fleet of K vehicles is used, each with a limited capacity of W. The fleet is based at a single *depot*, which also represents the disposal facility (dumpsite) where the vehicles are emptied. The time required to empty a vehicle is referred to as *dump time*.

Let us define a *feasible vehicle trip* as being a tour to and from the depot, including the service of some tasks within the vehicle's capacity. Thus, a *vehicle service* is a set of feasible vehicle trips, and its *workload time* is the time required to complete the vehicle's service, which includes the times for the service, deadheading and the dump. As a crew is assigned

to each vehicle, and bearing in mind that labour regulations impose a limit on their working period, a fixed workload time limit of L per vehicle is considered.

Given the aforementioned network of street segments and also the homogeneous fleet of K vehicles, the problem consists of determining a set of K sectors and a feasible vehicle service per sector, such that each task is serviced by one trip only, whilst minimising the following criteria: i) total routing time; ii) workload time imbalance, and; iii) the number of connected components in the subgraph induced by the tasks.

Total routing time (TT) measures the time required to perform all the trips in all the sectors, which equates to the sum over all the sectors of the workload times. It should thus be minimised, in order to reduce operational costs. *Workload time imbalance* (WIB), on the other hand, provides the difference between the workload time of the longest and the shortest sectors. This measure is related to fairness among the services provided by the vehicle crews, i.e., smaller values point to similar workload times that are assigned to different vehicle crews. Lastly, the *number of connected components* (CC) equals the number of connected components of each sector, over all sectors. Its minimisation increases connectivity, and thus it is used to pursue better designed solutions.

4. A two-phase heuristic: MTP

In this section we present the two-phase heuristic MTP. MTP's aim is to partition the graph into a given number of sectors, and then to build the trips within each sector, whilst minimising total routing time.

The MTP is a modified version of the TPH two-phase heuristic that is suggested by Mourão et al. (2009). TPH proved to be an efficient heuristic, if total routing time and workload time imbalance are the characteristics in focus. However, for practical applications, such as the one under study in this paper, other sector features should also be considered, such as their connectivity. Therefore, the purpose of MTP is to fill this gap.

Both TPH and MTP share the following sequence. In Phase 1, known as the sectoring phase, all the sectors are initialised as empty sets, and then each task is iteratively assigned to one sector. In Phase 2, which is the routing phase, those vehicle trips that minimise total routing time are identified. The difference between TPH and MTP relies on Phase 1, namely by the way in which the tasks are assigned to sectors.

4.1. Phase 1: sectoring

To partition the graph into sectors, the MST heuristic is proposed, as this is designed to promote solutions in which the demand graph within each sector is connected, whilst minimizing total routing time. MST is a modified version of the single task heuristic (STH) of Mourão et al. (2009). Next, the common parts of both heuristics are first described, and then their differences are highlighted.

Let us define an *open sector* as being a sector that remains available for expansion, and is otherwise *closed*. The process starts by defining K empty and open sectors. Then, K seed-tasks are selected by means of a seed-task selection rule that spreads the seeds over the entire network, and each one is assigned to a different sector.

An iterative process is then applied for the assignment of the remaining tasks to sectors. For each iteration, the open sector k with the smallest workload time estimate is selected for expansion. This workload time estimate, which is given by $WE(k)$, is based on preliminary trips, computed via the best insertion position for each task. Afterwards, a single unassigned task is selected, according to a task selection rule for expanding sectors, and the state of the sectors (open or closed) is then updated. The iterative process is repeated until all the tasks are assigned. It should be noted that sectors are built in parallel, in order to promote their workload time balance.

Both task selection rules – the seed-task selection and the task selection for expanding sectors – make use of a time distance measure, and thus the U-distance proposed in Mourão et al. (2009, Sec. 2.3) is used. For each pair of tasks (u, v), let D_{uv} be the distance from u to v, computed by the shortest deadheading time distance from u to v, excluding the deadheading times of u and v. As the base network is mixed, and given that each required edge is represented by two linked inverse arcs, u and $inv(u) = (j, i)$, the U-distance U_{uv} is computed by the minimum among at most eight shortest deadheading distances, as follows: $U_{uv} = \min\{D_{u,v};\ D_{u,inv(v)};\ D_{inv(u),v};\ D_{inv(u),inv(v)};\ D_{v,u};\ D_{v,inv(u)};\ D_{inv(v),u};\ D_{inv(v),inv(u)}\}$.

To pursue the goal of sectors connectivity, MST modifies: i) the task selection rule to expand a sector, and; ii) the criteria for closing a sector.

More precisely, at each iteration of MST, the task closest to the seed-task of sector k is selected among the unassigned tasks that share a node with at least one of the tasks already assigned to sector k. If such a task exists, then the assignment becomes definitive, and $WE(k)$ is updated. Otherwise, the sector is considered closed, as no more unassigned tasks could ensure its connectivity. It should be noted that, in this way, the sectors can all be closed before the complete assignment of all the tasks. This may happen, for instance, when the demand graph is not connected. In such cases, all sectors are reopened once again, and a new rule is defined to assign the remaining unassigned tasks to sectors: given an open sector k, the unassigned task closest to the seed-task of sector k is selected, and a new connected component of tasks is initialised for sector k.

With regards to STH, the unassigned task closest to the seed-task of sector k is selected (thus it does not have to share nodes with the tasks already assigned to the sector), and $WE(k)$ is recomputed. If $WE(k)$ does not exceed the time range L, then the task is assigned to sector k, otherwise sector k is considered closed, and the task remains unassigned.

A complete description of STH, as well as that of the time distance measure, the seed-task selection rule, and the workload time estimate, can be found in Mourão et al. (2009).

Due to the aforementioned differences, STH and MST promote solutions with different features. If the demand graph is connected, MST tends to generate solutions in which tasks within the same sector form a unique connected component, however it does not guarantee that the workload time estimate of each sector will not exceed the time range L. In turn, STH generates sectors with a workload time estimate that never exceeds L (although it may be exceeded later on, during the routing phase), but which may include several connected components per sector. Thus, whilst MST favours solutions with better connected sectors, STH tends to find solutions with a smaller workload time imbalance.

4.2. Phase 2: routing

In Phase 2, trips are determined by taking into account the assignments made during Phase 1, as in Mourão et al. (2009). For this purpose, the MCARP improvement merge (IM) heuristic suggested by Belenguer et al. (2006) is applied within each sector. The IM heuristic is an improvement of the extended augment merge (EAM) heuristic of Lacomme et al. (2004), which, in turn, extends to the MCARP the classical augment-merge heuristic of Golden and Wong (1981).

5. Local search heuristics

The two-phase heuristic MTP proposed in Section 4 has two major drawbacks: the range time constraints may be violated, which means that an infeasible solution can be identified; and the workload time imbalance may be too high (see Table 4 and Table 6, Section 6). To overcome these weaknesses, we devised a hill climbing heuristic HC, and a tabu search heuristic TS. Both HC and TS are tailored for minimising the imbalance, whilst maintaining the good features of the solution provided by MTP.

To pursue this objective, the evaluation of each incumbent solution S during the search process is computed by the following normalised weighted sum:

$$Eval(S) = \beta_{TT} * \frac{TT(S) - LB}{UB_{TT} - LB} + \beta_{CC} * \frac{CC(S) - K}{UB_{CC} - K} + \beta_{WIB} * \frac{WIB(S)}{UB_{WIB}} \quad (1)$$

The $Eval(.)$ function combines three measures: i) total routing time $TT(.)$; ii) total number of connected components $CC(.)$, and; iii) workload time imbalance $WIB(.)$. To ensure that all the measures have the same level of importance, the original values are scaled down to similar ranges, via lower bounds (LB and K), and upper bounds (UB_{TT}, UB_{CC}, and UB_{WIB}). LB represents the best-known lower bound for the MCARP (Belenguer et al., 2006; Gouveia et al., 2010), K is the given number of sectors, whereas UB_{TT}, UB_{CC}, and UB_{WIB} respectively measure the total routing time, the number of connected components, and the workload time imbalance of the initial solution.

Furthermore, each criterion is weighted by a $\beta \in \{0, 1\}$ parameter, namely β_{TT}, β_{CC}, and β_{WIB}, which allows the corresponding criterion to be, or not to be, considered for evaluating solutions. Better solutions thus present smaller values of $Eval(.)$.

Local search heuristics are designed to investigate solutions in the neighbourhood $N(S)$ of the current solution S, and then to move to the best neighbour in pursuit of the best solution. The neighbourhood of a solution is identified by means of small perturbations (moves) made over it. In the proposed local search heuristics, two different moves between a pair of tasks are considered: $Change(.)$ and $Swap(.)$. Given a solution S and the two tasks u and v, $Change(S, u, v)$ moves task v to the same trip as task u, and after it, whereas $Swap(S, u, v)$ exchanges the position of tasks u and v. The number of moves to be evaluated at each iteration increases with the number of tasks and sectors. Additionally, the evaluation of each neighbour solution obliges the re-computation of the number of connected components, with an associate high computational effort. To overcome this problem, only a subset of

moves is considered. The moves are stored in a restricted moves list (RML), similar to the restrictive candidate list that is proposed for the construction phase of the GRASP (Feo and Resende, 1995).

Given a solution S, consider all pairs of tasks (u, v) such that u and v belong to different sectors. For these pairs, let D_{max} and D_{min} be the maximum and the minimum of the D_{uv} distances, respectively. Then, $RML(S)$ includes the pairs of tasks (u, v) by non-decreasing order of the D_{uv} distances, such that $D_{uv} \leq D_{min} + \delta \times (D_{max} - D_{min})$, with $\delta \in [0, 1]$. However, even with this RML list, each iteration can be very time consuming. Therefore, a first improvement strategy is adopted, which means that the current solution is moved to a neighbour solution as soon as an improving solution is found.

As is observed with the computational results, the drawbacks of MST can be repaired by HC and TS, owing to their ability to provide feasible solutions with better balanced sectors. Next, Sections 5.1 and 5.2 respectively provide the description of HC and TS.

5.1. A hill climbing heuristic: HC

The HC is a hill climbing heuristic, which means that a neighbouring solution is accepted, only if it is better than the current one. The iterative search process is repeated until a maximum number of iterations ($MaxIt$) is reached, or until all neighbour solutions of the current one are non-improving solutions. The output solution is the best one found during the search process. Algorithm 1 summarises the general framework of HC.

Algorithm 1 The HC algorithm

Input: S ▷ Input solution
Input: $MaxIt$ ▷ Maximum number of iterations
$Cost_S \leftarrow Eval(S)$
$It \leftarrow 1$ ▷ Counter for iterations
repeat
  Create $RML(S)$
  $Cost_{BestN} \leftarrow +\infty$ ▷ Initialises the cost of the best neighbour solution
  repeat
    Remove the first pair of tasks (u, v) from $RML(S)$
    $NCh(S) \leftarrow Change(S, u, v)$ ▷ Apply the change move
    $NSw(S) \leftarrow Swap(S, u, v)$ ▷ Apply the swap move
    if ($NCh(S)$ is feasible **and** $Eval(NCh(S)) < Cost_{BestN}$) **then**
      $Cost_{BestN} \leftarrow Eval(NCh(S))$, $BestN \leftarrow NCh(S)$
    end if
    if ($NSw(S)$ is feasible **and** $Eval(NSw(S)) < Cost_{BestN}$) **then**
      $Cost_{BestN} \leftarrow Eval(NSw(S))$, $BestN \leftarrow NSw(S)$
    end if
  until ($RML(S)$ is empty **or** $Cost_{BestN} < Cost_S$)
  $FlagMove \leftarrow true$
  if $Cost_{BestN} < Cost_S$ **then**
    $S \leftarrow BestN$, $Cost_S \leftarrow Cost_{BestN}$
    $FlagMove \leftarrow false$
  end if
  $It \leftarrow It + 1$
until ($FlagMove$ **or** $It > MaxIt$)
Output: S ▷ Best solution found

5.2. A tabu search heuristic: TS

As is usually the case with this class of methods, the major drawback of the HC proposed here, is that it can easily be trapped in a local optimum. This problem can be diminished through applying more sophisticated methodologies, such as tabu search (Glover and Laguna, 1997). Tabu search uses an adaptive memory to escape from local optima. It seeks better solutions by allowing non-improving moves through an adaptive memory called *tabu list*. Algorithm 2 summarises the general framework of the TS algorithm.

Algorithm 2 The TS algorithm

Input: S, $TabTen$, $MaxIt$, $MaxImpIt$ ▷ Input solution, tabu tenure, and stopping criteria
$It, IterImp \leftarrow 1$ ▷ Counters for iterations
$TabuList = \{\}$
$Cost_S, Cost_{Best} \leftarrow Eval(S)$
$Best \leftarrow S$
repeat
  Create $RML(S)$
  $Cost_{BestN} \leftarrow +\infty$
  repeat
    Remove the first pair of tasks (u, v) from $RML(S)$
    $NCh(S) \leftarrow Change(S, u, v)$ ▷ Apply the change move
    $NSw(S) \leftarrow Swap(S, u, v)$ ▷ Apply the swap move
    if $NCh(S)$ is feasible **then**
      if $(Change(S, u, v) \notin TabuList$ **and** $Eval(NCh(S)) < Cost_{BestN})$ **then**
        $Cost_{BestN} \leftarrow Eval(NCh(S))$, $BestN \leftarrow NCh(S)$
      else if $(Change(S, u, v) \in TabuList$ **and** $Eval(NCh(S)) < Cost_{Best})$ **then**
        if $(Eval(NCh(S)) < Cost_{BestN})$ **then**
          $Cost_{BestN} \leftarrow Eval(NCh(S))$, $BestN \leftarrow NCh(S)$
        end if
      end if
    end if
    if $NSw(S)$ is feasible **then**
      if $(Swap(S, u, v) \notin TabuList$ **and** $Eval(NSw(S)) < Cost_{BestN})$ **then**
        $Cost_{BestN} \leftarrow Eval(NSw(S))$, $BestN \leftarrow NSw(S)$
      else if $(Swap(S, u, v) \in TabuList$ **and** $Eval(NSw(S)) < Cost_{Best})$ **then**
        if $(Eval(NSw(S)) < Cost_{BestN})$ **then**
          $Cost_{BestN} \leftarrow Eval(NSw(S))$, $BestN \leftarrow NSw(S)$
        end if
      end if
    end if
  until ($RML(S)$ is empty **or** $Cost_{BestN} < Cost_S$)
  $FlagMove \leftarrow true$
  if $Cost_{BestN} < +\infty$ **then**
    $S \leftarrow BestN$, $Cost_S \leftarrow Cost_{BestN}$
    $FlagMove \leftarrow false$
    $It \leftarrow It + 1$, $IterImp \leftarrow IterImp + 1$
    Update $TabuList$
    if $Cost_{BestN} < Cost_{Best}$ **then**
      $Best \leftarrow BestN$, $Cost_{Best} \leftarrow Cost_{BestN}$
      $IterImp \leftarrow 1$
    end if
  end if
until ($FlagMove$ **or** $It > MaxIt$ **or** $IterImp > MaxImpIt$)
Output: $Best$ ▷ Best solution found

In the proposed TS, the tabu status is applied to tasks, and a *tabu tenure* ($TabTen$) is set to define how long these tasks will remain with a tabu status. Moreover, moves involving tabu tasks can only be accepted if they lead to solutions that are better than the best one already found (aspiration criterion).

The stopping criteria are a maximum number of iterations ($MaxIt$), a maximum number of consecutive iterations without improvement ($MaxImpIt$), or no neighbour solutions are available.

6. Computational experiments

The proposed heuristics were coded in Delphi 7, and run on a 2.9 GHz Intel CORE i7-3520M CPU with 8 GB RAM.

Computational experiments were performed on two sets of instances. The first set, hereafter named as *slpr*, is based on the *lpr* instances introduced by Belenguer et al. (2006) as random MCARP instances that mimic street networks. These *lpr* instances were then transformed into SARP ones by adding two parameters: the number of sectors (K); and the maximum workload within each sector (L), which is considered fixed and equal to $21,600$ seconds. Moreover, it is assumed that a fleet of homogeneous vehicles, with capacity (W) equal to $10,000$ kilograms, is available at the depot node.

Table 1 lists the characteristics of the 15 *slpr* instances, namely the number of nodes, links, required edges, required arcs and sectors. A more detailed description of these instances can be found in Mourão et al. (2009).

Table 1: Characteristics of the *slpr* instances.

	number of				
instance	nodes	links	required edges	required arcs	sectors
slpr-a-01	28	94	0	52	2
slpr-a-02	53	169	5	99	2
slpr-a-03	146	469	33	271	4
slpr-a-04	195	651	34	469	7
slpr-a-05	321	1056	58	748	12
slpr-b-01	28	63	5	45	2
slpr-b-02	53	117	9	92	2
slpr-b-03	163	361	26	279	5
slpr-b-04	248	582	8	493	8
slpr-b-05	401	876	37	764	13
slpr-c-01	28	52	39	11	2
slpr-c-02	53	101	77	23	2
slpr-c-03	163	316	241	61	6
slpr-c-04	277	604	362	142	9
slpr-c-05	369	841	387	416	14

The second set, named as *seix*, includes nine instances – $seix1$ to $seix9$–, generated from a refuse collection network in Seixal, a municipality near Lisbon, Portugal. These instances are all derived from the same street network and differ in the number of sectors, vehicle capacity, and workload time limit, as depicted in Table 2.

To evaluate the performance of the proposed heuristics, total routing time (TT) in seconds, the number of connected components (CC), workload imbalance (WIB) in seconds, and the cpu running time ($tcpu$) in seconds, are all presented. For sake of simplicity, and if applicable, we also provide some percentage gaps for the total routing time, namely: LB_{gap}, and UB_{gap}. The lower bound gap is defined as $LB_{gap} = \frac{TT(H)-LB}{LB} \times 100\%$, where LB accounts for the best-known MCARP lower bound (Belenguer et al., 2006; Gouveia et al., 2010), and $TT(H)$ for the total routing time of the best feasible solution provided

Table 2: Characteristics of the *seix* instances.

instance	number of nodes	links	required edges	required arcs	sectors	time limit	vehicle capacity
seix1	106	214	84	52	2	18000	3000
seix2	106	214	84	52	2	18000	5000
seix3	106	214	84	52	2	18000	10000
seix4	106	214	84	52	3	14400	2000
seix5	106	214	84	52	3	14400	3000
seix6	106	214	84	52	3	14400	5000
seix7	106	214	84	52	4	10800	1000
seix8	106	214	84	52	4	10800	2000
seix9	106	214	84	52	4	10800	3000

by one of the local search heuristics (HC or TS). In turn, the upper bound gap is given by $UB_{gap} = \frac{TT(MTP)-TT(H)}{TT(H)} \times 100\%$, with $TT(MTP)$ being the total routing time obtained through the two-phase heuristic MTP.

6.1. Parameters tuning

The parameters for HC and TS were tuned via a computational study. The reasoning behind this study was to find the best set of parameters, i.e. the one that provides the best average results for all the criteria evaluated. For this purpose, *slpr* instances were used. Moreover, the maximum number of iterations ($MaxIt$), and, if applicable, the maximum number of iterations without improvement ($MaxImpIt$), and tabu tenure ($TabTen$) were respectively set to $1,000$; 100 and 7. By varying $\delta \in \{0, 0.5, 1\}$ and the β parameters (β_{TT}, β_{CC}, $\beta_{WIB} \in \{0, 1\}$), a total of 21 distinct settings were tested.

This first experiment allowed us to conclude that: i) it is sufficient to limit $MaxIt$, and $MaxImpIt$ to 600 and 40 iterations respectively; ii) $\delta = 0$ always provides the worst results, whereas $\delta = 1$ is quite time consuming, with average results sometimes worse than for $\delta = 0.5$, and; iii) the inclusion of all the criteria, by setting the β parameters to 1, is mandatory for the good performance of the methods.

Accordingly, with the aforementioned conclusions ($\delta = 0.5$, $MaxIt = 600$, and $MaxImpIt = 40$), some tuning tests were conducted for the $TabTen$ parameter, namely setting it to 5, 7, and 9. The average results did not highlight any significant differences, and to some extent they were inconclusive as to which $TabTen$ value performs better.

The aforementioned experiments brought up the subject of the influence of β parameters on the performance of the algorithms. To study the impact of these parameters, we run both algorithms with the setting $MaxIt = 600$, $\delta = 0.5$, and, if applied, $MaxImpIt = 40$ and $TabTen = 7$. The results are reported in Table 3. The first two columns identify the algorithm (HC or TS) and the type of values (Min, Avg, and Max respectively for the minimum, the average, and the maximum value). Then there are three groups of four columns, with each group representing one criterion and each column a scenario for the β parameters. The columns are labelled as a triple, representing the values assigned to β_{TT}, β_{CC}, and β_{WIB} respectively, which, in turn, indicate whether the corresponding criterion is being considered (1), or not (0).

The results listed in Table 3 show that the best LB_{gap} gaps and the best WIB values are obtained if the corresponding β parameter is set to one, and the remaining ones are set

Table 3: Algorithm's performance for different β parameter settings – *slpr* instances.

		LB_{gap} (%) $\frac{TT(H)-LB}{LB}$				connected components CC				workload imbalance $WIB(sec)$			
		(100)	(010)	(001)	(111)	(100)	(010)	(001)	(111)	(100)	(010)	(001)	(111)
HC	Min	**0.90**	1.53	2.92	2.11	**2**	**2**	**2**	**2**	522	39	**0**	3
	Avg	**3.60**	4.83	7.52	5.08	15	**6**	20	**6**	5057	3244	**19**	1056
	Max	**6.89**	9.21	19.44	8.89	42	**14**	93	**14**	12018	10297	**93**	6133
TS	Min	**0.54**	1.53	4.41	2.11	**2**	**2**	**2**	**2**	560	39	**0**	5
	Avg	**3.30**	4.86	8.58	4.98	16	**6**	22	**6**	5594	3320	**8**	797
	Max	**6.45**	9.21	19.44	8.77	56	**14**	93	**14**	12645	12465	**25**	5828

to zero, no matter which algorithm is being considered. Regarding the number of connected components, the best results are provided by the scenarios in which $\beta_{CC} = 1$. Thus, it seems crucial to take into account the number of connected components during the solution's evaluation, no matter what are the values considered for the other β parameters. To sum up, scenario (111) seems to be a good commitment, as by considering all the criteria simultaneously, the obtained solutions do not appear to deteriorate the quality of each of the individual criterion too much.

In the next two sections we provide a more detailed analysis of the algorithms' performance on each one of the instance sets. Henceforward, all the reported results were obtained with the following setting: $MaxIt = 600$, $\delta = 0.5$, $(\beta_{TT}, \beta_{CC}, \beta_{WIB}) = (1, 1, 1)$, and, with respect to TS, $MaxImpIt = 40$ and $TabTen = 7$.

6.2. Analysis for the slpr instances

Table 4 reports the results, per instance, obtained with MTP and with both local search algorithms (HC and TS). The first column identifies the instance, and the remaining seventeen columns are divided into six groups. The groups, in turn, are subdivided into columns, headed as MTP, HC, and TS, to identify the algorithm they refer to. The first three groups provide results about total routing time. The first group, whose heading is $TT(sec)$, lists the total routing time, which is expressed in seconds, and the second and third groups list the LB_{gap}, and the UB_{gap} (see Section 6). It is worth noting that positive UB_{gap} percentage gaps highlight the fact that local search algorithms were able to decrease the total routing time provided by the MTP algorithm. The fourth group, named as $CC - K$, compares the number of connected components (CC) with the number of sectors (K), while the fifth group lists workload imbalance $WIB(sec)$, expressed in seconds. The last group reports CPU time, in seconds. It is worth noting that the time for both HC and TS also includes the time for determining the initial solution, which is the running time of the MTP algorithm.

Analysing Table 4, it can be stated that:

- Despite being computed via a MCARP lower bound, which can be a weak lower bound for the problem at hand, the LB_{gap} percentage gaps are relatively small, which points to good feasible solutions. The smallest TT values are usually provided by the MTP, as may be confirmed by the negative UB_{gap}. Only in 4 out of 15 instances (a-05, b-01, b-03, b-05) does TS yield the best value, being outperformed by MTP in the remaining ones. In turn, the HC algorithm performs worse than MTP in all but one instance

Table 4: Computational results – *slpr* instances.

	total routing time																
	$TT(sec)$			LB_{gap} $\frac{TT(H)-LB}{LB}$			UB_{gap} $\frac{TT(MTP)-TT(H)}{TT(H)}$		$CC-K$			$WIB(sec)$			cpu time (sec)		
instance	MTP	HC	TS	MTP	HC	TS	HC	TS	MTP	HC	TS	MTP	HC	TS	MTP	HC	TS
slpr-a-01	13795	13855	13855	2.31	2.75	2.75	-0.43	-0.43	0	0	0	39	7	7	0.03	0.11	0.81
slpr-a-02	29423	29495	29495	4.89	5.14	5.14	-0.24	-0.24	0	0	0	413	15	15	0.09	0.53	2.68
slpr-a-03	78789	79178	79243	3.51	4.02	4.11	-0.49	-0.58	3	2	1	360	14	13	0.22	8.99	45.85
slpr-a-04	134790	135156	134990	6.18	6.47	6.34	-0.27	-0.15	0	0	0	6441	154	107	0.26	26.43	164.42
slpr-a-05	217571	217724	217437	7.32	7.39	7.25	-0.07	0.06	0	0	0	8402	5598	5465	0.43	134.61	1903.20
slpr-b-01	15327	15387	15256	3.32	3.72	2.84	-0.39	0.46	0	0	0	1165	3	14	0.02	0.09	0.59
slpr-b-02	29752	29754	29754	3.83	3.84	3.84	-0.01	-0.01	0	0	0	216	6	6	0.04	0.47	2.95
slpr-b-03	82790	83067	82745	6.33	6.69	6.28	-0.33	0.05	0	0	0	2917	67	37	0.14	7.00	50.90
slpr-b-04	137775	138215	138060	8.54	8.89	8.77	-0.32	-0.21	0	0	0	4163	116	113	0.25	20.30	135.86
slpr-b-05	229114	227991	226620	9.21	8.68	8.02	0.49	1.09	0	0	0	10297	6133	5828	0.43	77.30	2020.20
slpr-c-01	18925	19033	19033	1.53	2.11	2.11	-0.57	-0.57	0	0	0	343	5	5	0.02	0.33	1.40
slpr-c-02	37382	37559	37559	2.87	3.36	3.36	-0.47	-0.47	0	0	0	376	17	17	0.05	1.44	6.52
slpr-c-03	116075	116204	116955	4.46	4.58	5.25	-0.11	-0.76	0	0	0	4310	3371	157	0.17	24.69	130.10
slpr-c-04	174432	174554	174490	3.56	3.63	3.59	-0.07	-0.03	0	0	0	4588	137	90	0.26	72.99	592.86
slpr-c-05	270028	270732	270745	4.71	4.98	4.98	-0.26	-0.27	0	0	0	4386	198	74	0.42	161.27	2071.76

(b-05). Nevertheless, neither HC nor TS significantly increase the total routing time, as the smaller UB_{gap} percentage gaps are -0.57% and -0.76%, respectively.

- In all but one instance (a-03), the number of connected components matches the number of sectors, no matter which algorithm is applied. For the a-03 instance, TS outperforms both the MTP and HC algorithms. It is worth remarking that this instance is the only one that has a demand subgraph with two connected components, and that one of these components contains only a few number of tasks;

- With regards to the WIB criterion, the results show that both HC and TS provide significantly better solutions than does MTP. Moreover, in 9 instances, TS outperforms HC, which, in turn, is only better for b-01;

- As expected, running time increases with the dimension of the problem, no matter which method is being considered. The MTP runs in a few seconds, whereas the local search based algorithms are more time consuming. The differences between HC and TS cpu times are certainly due to the number of iterations required until one of the stopping criteria is reached, and also to the fact that each iteration obliges re-computing the number of connected components, which is very time consuming. However, the time is always less than 35 minutes, which is meaningless for the problem being solved.

To conclude, the results depicted in Table 4 indicate that MTP provides good feasible solutions with respect to total routing time (TT) and to the number of connected components (CC), but with high workload time imbalance (WIB). The local search algorithms are able to decrease the workload time imbalance, with small increases in total routing time, whilst maintaining, or even improving, the number of connected components. This tendency to increase total routing time, whilst decreasing the imbalance, is quite intuitive, as imposing a smaller imbalance can be expressed as being an additional constraint to the problem. Leaving aside running time, which is meaningless for the problem being analysed, TS is the local search algorithm that generally provides the best results.

To assess the efficiency of TS over the alternative approaches already reported in the literature, namely CTH, STH and BIH (see Mourão et al., 2009), Table 5 lists the difference between the results provided by TS and by the alternative approaches for each one of the criteria being evaluated. Thus, negative values mean that TS performs better than the alternative approach. The results are divided into three groups, one for each criterion. In Mourão et al. (2009) the number of connected components is not reported. Thus we had to compute them by re-running the three aforementioned approaches: CTH, STH and BIH.

In Table 5 it can be seen that TS yields not only the smallest workload imbalances in all but 2 instances (a-05 and b-05), but also the lowest number of connected components in all the cases. In fact, the decrease of the CC criterion is quite significant for the large-sized instances, whilst CTH, STH, or BIH equal the CC value obtained by TS only for few of the small-sized instances. Furthermore, these improvements are generally accompanied by a decrease in total routing time for the large-sizes instances, whilst an increase is observed for

Table 5: Comparing TS with previous approaches – *slpr* instances.

	$TT(sec)$			CC			$WIB(sec)$		
instance	TS–CTH	TS–STH	TS–BIH	TS–CTH	TS–STH	TS–BIH	TS–CTH	TS–STH	TS–BIH
slpr-a-01	174	94	110	-1	0	0	-26	-172	-138
slpr-a-02	222	246	-248	-2	-2	-2	-126	-270	-82
slpr-a-03	619	732	-835	-5	-4	-4	-452	-698	-383
slpr-a-04	680	-645	-2341	-20	-18	-18	-572	-777	-411
slpr-a-05	-3024	-4961	-7279	-35	-26	-26	4523	4712	4957
slpr-b-01	69	-64	49	-1	-2	-2	-139	-180	-93
slpr-b-02	118	10	-487	0	0	0	-786	-26	-135
slpr-b-03	200	-704	-1352	-10	-5	-5	-677	-142	-198
slpr-b-04	2206	-1174	-2681	-22	-15	-15	-818	-748	-299
slpr-b-05	-3710	-12072	-15427	-65	-83	-83	2850	4987	5324
slpr-c-01	154	8	19	0	0	0	-234	-208	-103
slpr-c-02	280	428	101	0	0	0	-248	-196	-245
slpr-c-03	282	395	-1066	-6	-10	-10	-488	-193	-339
slpr-c-04	-1317	-1453	-3735	-10	-21	-21	-623	-923	-640
slpr-c-05	-1562	-1785	-6082	-20	-31	-31	-827	-583	-244

the small-sized ones. With regards to instances a-05 and b-05, total routing time decreases and the number of connected components is substantially smaller, despite the increase of the WIB criteria, which indicates solutions that possess better characteristics from a practical point of view.

From a more managerial perspective, we can conclude that TS provides better quality solutions than the previous approaches. In general, sectors are better balanced, which reveals an improvement on fairness amongst crews. Furthermore, a smaller number of connected components point to sectors that are geographically located in more delimited areas, which is also a benefit for waste collection management.

6.3. Analysis for the seix instances

Table 6 summarises the results, per Seixal instance, yielded by MTP, HC, and TS.

As can be observed, MTP always provides the best total routing time. Nevertheless, the UB_{gap} percentages are higher than -0.68 in all but one instance ($seix7$), which indicates acceptable increases in the TT values. The differences between HC and TS are meaningless: TS is better than HC in 4 out of 9 instances, the largest difference being 20 seconds; and is worse in 3 cases, the largest difference being 67 seconds. All the methods reach the minimum number of connected components, which is the number of sectors. Both local search methods obtain solutions with much smaller workload imbalances than that of MTP. These differences are particularly significant for the instances with 4 sectors ($seix7 - seix9$), in which the WIB decreases from about more than 50 minutes, to less than 5 minutes. Taking into account all the three criteria simultaneously (TT, CC and WIB) TS performs slightly better than HC. With regards to CPU times, MTP is the quickest method, whereas TS is the slowest one.

To conclude this analysis, Table 7 compares TS with the previous alternative approaches, namely STH, BIH and CTH.

As has already been stated for the *slpr* instances, TS usually gives worse TT values than the previous approaches. Nevertheless, this increase is well compensated with better imbalances and a smaller number of connected components. Thus, once again, it can be stated that TS outperforms the previous approaches, as it produces solutions that are better fitted for real life applications.

Table 6: Computational results – *seix* instances.

	total routing time																
	$TT(sec)$			LB_{gap} $\frac{TT(H)-LB}{LB}$			UB_{gap} $\frac{TT(MTP)-TT(H)}{TT(H)}$		$CC-K$			$WIB(sec)$			cpu time (sec)		
instance	MTP	HC	TS	MTP	HC	TS	HC	TS	MTP	HC	TS	MTP	HC	TS	MTP	HC	TS
seix1	32423	32557	32537	4.94	5.37	5.31	-0.41	-0.35	0	0	0	1691	31	11	0.02	1.58	8.13
seix2	31549	31618	31618	3.58	3.80	3.80	-0.22	-0.22	0	0	0	1179	108	108	0.00	1.40	8.60
seix3	30717	30804	30797	2.18	2.46	2.44	-0.28	-0.26	0	0	0	1367	40	33	0.00	1.45	8.64
seix4	33579	33606	33673	5.41	5.49	5.70	-0.08	-0.28	0	0	0	1460	81	23	0.00	2.43	11.95
seix5	32818	32872	32910	6.22	6.39	6.52	-0.16	-0.28	0	0	0	1047	40	4	0.00	2.47	14.54
seix6	31472	31581	31581	3.32	3.68	3.68	-0.35	-0.35	0	0	0	1239	25	25	0.00	2.45	13.78
seix7	37513	38295	38287	6.87	9.10	9.07	-2.08	-2.06	0	0	0	3775	69	3	0.00	2.43	16.46
seix8	33809	34006	34039	6.13	6.75	6.85	-0.58	-0.68	0	0	0	3003	232	17	0.00	4.62	22.26
seix9	33003	33197	33177	6.82	7.44	7.38	-0.59	-0.53	0	0	0	3510	102	102	0.00	3.14	11.06

Table 7: Comparing TS with previous approaches – *seix* instances

	$TT(sec)$			CC			$WIB(sec)$		
instance	TS–CTH	TS–STH	TS–BIH	TS–CTH	TS–STH	TS–BIH	TS–CTH	TS–STH	TS–BIH
seix1	558	937	371	-2	-4	-4	-56	-195	-56
seix2	29	48	-250	-2	-4	-4	-171	-68	-171
seix3	60	59	-52	-2	-4	-4	-122	21	-122
seix4	497	306	399	-4	-2	-2	-431	-689	-431
seix5	69	66	-362	-3	-2	-2	-270	-304	-270
seix6	157	70	266	-3	-4	-4	-133	-234	-133
seix7	1060	1248	1548	-2	-2	-2	-753	-948	-753
seix8	-66	8	-399	-3	-3	-3	-319	-381	-319
seix9	1121	1042	1055	-4	-4	-4	-260	-138	-260

7. Conclusions

In this paper we propose a constructive heuristic and two local search heuristics for a sectoring-arc routing problem. Using a real life application, namely the collection of urban waste, the proposed methods were designed to provide solutions with reduced total routing time but also that incorporate some required practical features, such as sectors that are both balanced and connected.

Each of the proposed methods were evaluated for two sets of instances: benchmark instances that were randomly generated according to real life applications, and instances derived from real data. The constructive heuristic (MTP) proved to be very fast and was generally able to provide solutions with the optimal number of connected components. Nevertheless, it tends to produce solutions with high imbalance values, mainly for large-sized instances.

The local search heuristics, one being a hill climbing approach (HC) and the other a tabu search approach (TS), were devised to simultaneously handle the three optimisation criteria. The computational results highlight the importance of considering all the criteria for evaluating solutions during the search process: if only one criterion is considered, then the quality of the solution increases for the criterion that is being considered, but decreases in the majority of cases for the other two criteria.

From a more managerial perspective, combining MTP with TS yields significant practical improvements. In fact, sectors are better balanced, with a lower number of connected components, which in turn allows for the identification of sectors which are geographically located in more delimited areas, which thus simplifies waste collection management.

The design of the sectors plays a crucial role when devising solutions for the problem being handled. Therefore, one future research direction is to developed more tailored heuristics or methaheuristics that iteratively alternate between the design of the sectors and the design of the vehicle trips.

Acknowledgement This work is supported by National Funding from the FCT-Fundação para a Ciência e a Tecnologia, under the PEsT-OE/MAT/UI0152 and PTDC/EGE-GES/121406/2010 projects.

Assis, L. S., Franca, P. M., Usberti, F. L., 2014. A redistricting problem applied to meter reading in power distribution networks. Computers & Operations Research 41, 65–75.

Bação, F., Lobo, V., Painho, M., 2005. Applying genetic algorithms to zone design. Soft Computing 9 (5), 341–348.

Belenguer, J. M., Benavent, E., Lacomme, P., Prins, C., 2006. Lower and upper bounds for the mixed capacitated arc routing problem. Computers & Operations Research 33 (12), 3363–3383.

Bozkaya, B., Erkut, E., Laporte, G., 2003. A tabu search heuristic and adaptive memory procedure for political districting. European Journal of Operational Research 144 (1), 12–26.

Cattrysse, D., Van Oudheusden, D., Lotan, T., 1997. The problem of efficient districting. OR Insight 10 (4), 9–13.

Constantino, M., Gouveia, L., Mourão, M. C., Nunes, A. C., 2015. The mixed capacitated arc routing problem with non-overlapping routes. European Journal of Operational Research doi: 10.1016/j.ejor.2015.01.042.

Corberán, A., Laporte, G. (Eds.), 2014. Arc Routing: Problems, Methods, and Applications. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia.

Corberán, A., Prins, C., 2010. Recent results on arc routing problems: An annotated bibliography. Networks 56 (1), 50–69.

Dror, M. (Ed.), 2000. Arc Routing: Theory, Solutions and Applications. Kluwer Academic Publishers, Dordrecht.

Feo, T., Resende, M., 1995. Greedy randomized adaptive search procedures. Journal of Global Optimization 6 (2), 109–133.

Ghiani, G., Laporte, G., 2001. Location-arc routing problems. OPSEARCH 38 (2), 151–159.

Glover, F., Laguna, M., 1997. Tabu Search, 1st Edition. Kluwer Academic Publishers: Dordrecht, Netherlands.

Golden, B., Raghavan, S., Wasil, E. (Eds.), 2008. The Vehicle Routing Problem: Latest Advances and New Challenges. Operations Research/Computer Science Interfaces Series. Springer.

Golden, B. L., Wong, R. T., 1981. Capacitated arc routing problems. Networks 11 (3), 305–315.

Gonzalez-Ramírez, R. G., Smith, N. R., Askin, R. G., Miranda, P. A., Sánchez, J. M., 2011. A hybrid metaheuristic approach to optimize the districting design of a parcel company. Journal of Applied Research and Technology 9 (1), 19–35.

Gouveia, L., Mourão, M. C., Pinto, L. S., 2010. Lower bounds for the mixed capacitated arc routing problem. Computers & Operations Research 37 (4), 692–699.

Haugland, D., Ho, S. C., Laporte, G., 2007. Designing delivery districts for the vehicle routing problem with stochastic demands. European Journal of Operational Research 180 (3), 997–1010.

Jarrah, A. I., Bard, J. F., 2012. Large-scale pickup and delivery work area design. Computers & Operations Research 39 (12), 3102–3118.

Jozefowiez, N., Semet, F., Talbi, E.-G., 2009. An evolutionary algorithm for the vehicle routing problem with route balancing. European Journal of Operational Research 195 (3), 761–769.

Kim, B. I., Kim, S., Sahoo, S., 2006. Waste collection vehicle routing problem with time windows. Computers & Operations Research 33 (12), 3624–3642.

Lacomme, P., Prins, C., Ramdane-Chérif, W., 2001. A genetic algorithm for the capacitated arc routing problem and its extensions. In: Boers, E., et al. (Eds.), Applications of Evolutionary Computing. Vol. 2037 of Lecture Notes in Computer Science. Springer, Berlin, pp. 473–483.

Lacomme, P., Prins, C., Ramdane-Chérif, W., Oct. 2004. Competitive memetic algorithms for arc routing problems. Annals of Operations Research 131 (1-4), 159–185.

Lei, H., Laporte, G., Guo, B., 2012. Districting for routing with stochastic customers. EURO Journal on Transportation and Logistics 1 (1–2), 67–85.

Lin, H. Y., Kao, J. J., 2008. Subregion districting analysis for municipal solid waste collection privatization. Journal of the Air & Waste Management Association 58, 104–111.

Mourão, M. C., Nunes, A. C., Prins, C., 2009. Heuristic methods for the sectoring arc routing problem. European Journal of Operational Research 196 (3), 856–868.

Mourgaya, M., Vanderbeck, F., 2007. Column generation based heuristic for tactical planning in multi-period vehicle routing. European Journal of Operational Research 183 (3), 1028–1041.

Muyldermans, L., Cattrysse, D., Van Oudheusden, D., 2003. District design for arc-routing applications. Journal of the Operational Research Society 54 (11), 1209–1221.

Oyola, J., Løkketangen, A., 2014. GRASP-ASP: An algorithm for the CVRP with route balancing. Journal of Heuristics 20 (4), 361–382.

Perrier, N., Langevin, A., Campbell, J. F., 2008. The sector design and assignment problem for snow disposal operations. European Journal of Operational Research 189 (2), 508–525.

Ramos, T. R. P., Oliveira, R. C., 2011. Delimitation of service areas in reverse logistics networks with multiple depots. Journal of the Operational Research Society 62 (7), 1198–1210.

Ríos-Mercado, R. Z., Fernández, E., 2009. A reactive GRASP for a commercial territory design problem with multiple balancing requirements. Computers & Operations Research 36 (3), 755–776.

Salazar-Aguilar, M. A., Ríos-Mercado, R. Z., González-Velarde, J. L., Molina, J., 2012. Multiobjective scatter search for a commercial territory design problem. Annals of Operations Research 199 (1), 343–360.

Salhi, S., Rand, G. K., 1989. The effect of ignoring routes when locating depots. European Journal of Operational Research 39 (2), 150–156.

Simchi-Levi, D., 1992. Hierarchical planning for probabilistic distribution-systems in euclidean spaces. Management Science 38 (2), 198–211.

Teixeira, J., Antunes, A. P., Sousa, J. P., 2004. Recyclable waste collection planning – a case study. European Journal of Operational Research 158 (3), 543–554.

Toth, P., Vigo, D. (Eds.), 2014. Vehicle Routing: Problems, Methods, and Applications, 2nd Edition. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia.

Wøhlk, S., 2008. A decade of capacitated arc routing. In: Golden, B., Raghavan, S., Wasil, E. (Eds.), The Vehicle Routing Problem: Latest Advances and New Challenges. Operations Research/Computer Science Interfaces. Springer, Berlin, pp. 29–48.