A Novel Model for Arc Territory Design: Promoting Eulerian Districts

Gabriela García-Ayala
Tecnologico de Monterrey, Mexico
mg_garcia@yahoo.com

José Luis González-Velarde
Tecnologico de Monterrey, Mexico
gonzalez.velarde@itesm.mx

Roger Z. Ríos-Mercado
Graduate Program in Systems Engineering
Universidad Autónoma de Nuevo León (UANL), Mexico
roger.rios@uanl.edu.mx

Elena Fernández
Statistics and Operations Research Department
Universitat Politècnica de Catalunya - BcnTech
e.fernandez@upc.edu

September 2013
Revised: 16 February 2015
Revised: 17 July 2015
Revised: 21 September 2015

Abstract

The problem of district design for the implementation of arc routing activities is addressed. The aim is to partition a road network into a given number of sectors to facilitate the organization of the operations to be implemented within the region. This problem arises in numerous applications such as postal delivery, meter readings, winter gritting, road maintenance, and municipal solid waste collection. An integer linear programming model is proposed where a novel set of node parity constraints to favor Eulerian districts is introduced. Series of instances were solved to assess the impact of these parity constraints on the objective function and deadhead distance. Networks with up to 401 nodes and 764 edges were successfully solved. The model is useful at a tactical level as it can be used to promote workload balance, compactness, deadhead distance reduction and parity in districts.

Keywords: Combinatorial optimization; Districting; Integer linear programming; Arc services.

1 Introduction

The problem of district design for the implementation of arc routing activities is addressed. The aim is to partition a road network into a given number of sectors to facilitate the organization of the operations to be implemented within the region. This problem arises in numerous applications such as postal delivery, meter readings, winter gritting, road maintenance, and municipal solid waste collection. A proper districting plan that divides the entire area into several balanced subregions promotes competition among contractors for arc routing services. Allowing more contractors to bid can reduce the investment risk and make it more attractive for companies to bid as well as prevent the domination of the service by one large company [12].

Districting decisions are made at a strategic or tactical management level, routing decisions are operational and made on a regular basis [8]. The subset of edges assigned to a depot constitutes a district. In contrast to the clear objectives in pure location or routing problems, it appears to be more difficult to define exact criteria for designing good districts for arc routing [15].

A general model is proposed where the focus is on the districting decisions such that the tactical or planning level decisions are not mixed with the operational decisions. Instead, we use criteria that should lead to the formation of good routing. Typical criteria for districting are (a) contiguity, (b) compactness, (c) deadhead distance and (d) work balance. A district is contiguous if it is possible to travel between any two points within the district without having to leave the district. A district is said to be compact if it is nearly round shaped or square, nondistorded, without holes, and has a smooth boundary [3]. Concentrating service activity in compact districts means shorter travel distances. Compactness is an intuitive measure for which several measures have been proposed in the districting literature [9] but none of these is comprehensive. Deadhead refers to the traveled distance where no service is to be performed. Workload balance refers to the degree in which every district is required to perform the same amount of work.

Compactness and contiguity lead to more efficient routing of vehicles [16]; however, the literature that takes deadhead into account at a strategic level is scarce. Deadhead distance is very hard to model at a strategic level because routing has to be done in order to take it into consideration. The model proposed in this paper takes deadhead into account at the strategic level by introducing the parity constraints.

Most of the work done in territory design/districting problems such as political districting, sales territory alignment, commercial territory design, health care districting, school district

design, and emergency services are node-based partitioning models, that is, the interest is on creating districts that are a partition of the set of vertices because the service is given at the nodes. In comparison with node-based districting, the problem of districting in connection with vehicle routing for collection or distribution services has received very little attention (Perrier et al. [17]). It should be noted that node- and edge-base districting models have different mathematical structure and therefore algorithms and methods developed for node-based districting models are not quite applicable to edge-based models. For excellent surveys on node-based districting the reader is referred to the works of Kalcsics et al. [10], Zoltners and Sinha [23], Duque et al. [6], and Ricca et al. [19]. Here, we highlight the most relevant work on edge/arc-based districting models.

Bodin and Levy [2] introduce the Arc Partitioning Problem, where arcs in a connected network are broken into a set of approximately equally weighted partitions, which is implemented in postal delivery. Campbel and Langevin [5, 4] develop sectorization models for snow removal and disposal. In [5], they introduce the model of assigning sectors to disposal sites. They develop a simple two-phase heuristic that it is applied to a real-world case in Montreal. Then, in [4], they extend their model to allow for disposal site location decisions as well. No solution method is given in this follow-up work. We must point out that in both of these works, no districting decisions are made, that is, the sectors are already fixed. Later, Perrier et al. [18] present a model and two heuristic solution approaches based on mathematical optimization for the problem of partitioning a road network into sectors and allocating sectors to snow disposal sites for snow disposal operations. Given a road network and a set of planned disposal sites, the problem is to determine a set of non-overlapping sub-networks, called sectors, according to several criteria related to the operational effectiveness and the geographical layout, and to assign each sector to a single snow disposal site so as to respect the capacities of the disposal sites, while minimizing relevant variable and fixed costs. Their approach uses single street segments as the units of analysis and they consider sector contiguity, sector balance and sector shape constraints, hourly and annual disposal site capacities, as well as single assignment requirements. The resulting model is based on a multi-commodity network flow structure to impose the contiguity constraints in a linear form. The two solution approaches were tested on data from the city of Montreal in Canada. Muyldermans et al. [15, 14] present a different approach for tackling arc districting problems. First in Muyldermans et al. [15], they address an arc districting problem for salt spreading operations. They present an ILP model that considers the following planning criteria: ability to support good routing, balance in workload, compactness of the districts, and centrality of

the depot. They present a heuristic procedure for the districting problem and its application to a real-world network in Antwerp. In their follow-up work [14], they present a framework for general arc districting problems considering contiguity as well. They also analyze cases where different objectives, such as minimizing number of vehicles may be preferred. The heart of their approach is the transformation of the given road graph into an Eulerian graph and then using elementary cycles as the main basic units. From this, the proposed heuristic seeks to form the cycles simultaneously by aggregating basic units to each district. Mourão et al. [13] address the sectoring arc routing problem, which consists of both deciding the arc partition and the vehicle routing in each district. It is a combination of two families of classical problems: sectoring problems and arc routing problems. Contiguity is not required in this work. The districts are built by optimizing routing costs. No compactness measure is considered in the model. A two phase heuristic is proposed. A pre-assignment of edges to depots is made at a first phase, and revised at a second phase along with vehicle routing. More recently, Butsch et al. [3] propose a heuristic for districting problems arising in an arc routing context. The aim is to find arc partitions that satisfy two hard criteria: complete and exclusive assignment as well as contiguity; and several soft criteria: balance, small deadheading, local compactness, and global compactness. To achieve this they use a weighted objective function containing the four soft criteria. The proposed heuristic applies a construction procedure followed by a tabu search improvement phase in which several subroutines are defined and selected according to a roulette wheel mechanism, as in adaptive large neighborhood search. Extensive tests conducted on instances derived from real-world street data confirm the efficiency of the proposed methodology. Silva de Assis et al. [22] address an edge redistricting problem arising in meter reading in power distribution networks They transformed their edge districting problem into a node-districting problem and propose a GRASP metaheuristic. In their appraoch they consider balancing and connectivity constraints and similarity with the existing districts.

As can be seen from the literature, previous works are either application specific or do not take into account all criteria (a) through (d), except for the work of Butsch et al. [3] who consider these criteria from a heuristic perspective. The aim of the present work is (i) to propose a new general model that includes all four criteria and introduces a new criterion as well: parity, and (ii) to derive an exact algorithm for solving the problem.

Parity is not a novel concept, and it has been known to affect routing decisions ever since the time of Euler. However, no models were found that take it into account when districting for routing services takes place. When the routing design is to take place, a Chinese Postman

Tour has to be found over the arcs in each partition. Thus each subgraph in the partition is desired to be as close to an Eulerian graph as possible. The necessary and sufficient condition for an Euler cycle to exist is that every node be of even degree. Parity is then defined as the criterion that penalizes arc partitions that induce odd degree of nodes in each subgraph of the partition. Adding such a criterion to the arc districting model will lead to partitions that are closer to an Eulerian graph, which will in turn allow for more efficient vehicle routing by reducing deadhead.

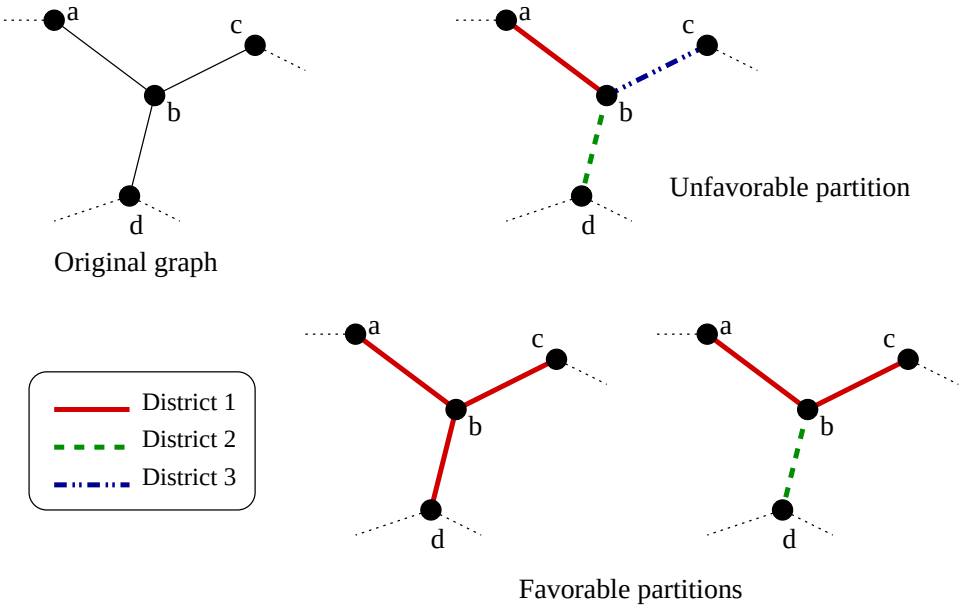


Figure 1: Possible partitions for a node of degree 3.

To illustrate this, suppose that the nodes in Figure 1 are part of a network to be partitioned into three districts. There are three arcs to be assigned. Note that since node b has odd degree in the original graph, there is no possible partition that will allow it to have even degree in every district. However, a constraint that favors the imparity to be maintained in only one of the districts, resulting in a favorable partition, can be built. An unfavorable partition would be a partition that assigns one edge to each district making then node b to be of odd degree in each of the districts.

In a similar manner, if a node has even degree in the original graph, a partition where it keeps its even degree among the districts should be preferred over partitions where it does not. Assume nodes in Figure 2 are part of a network to be partitioned into three districts. Cases of partitions where node c maintains the even degree are shown. These cases are considered favorable over partitions where node c has odd degree in two of the three districts.

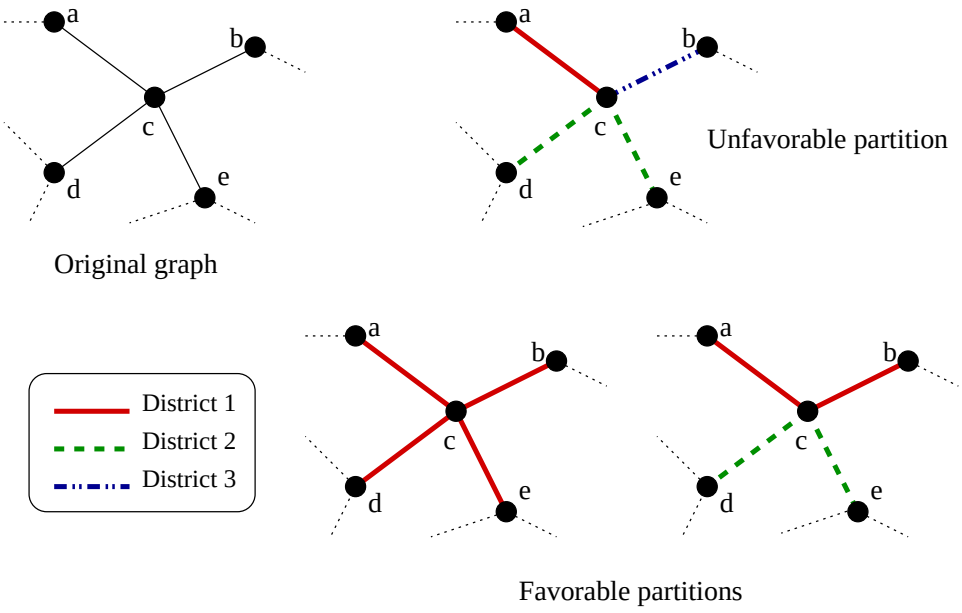


Figure 2: Possible partitions for a node of degree 4.

This criterion is novel and to the best of our knowledge has not been seen in any of the reviewed literature at the district design level. Odd degree nodes in a partition translate into deadhead time. A service vehicle will have to travel one edge of an odd degree node twice: once where the service will be delivered, and once without service in order to get back to the next edges. In this paper we propose this new parity criterion and a general model for arc districting which considers all typical criteria (a) through (d).

The remainder of this paper is organized as follows. Section 2 states the addressed problem and describes the proposed integer linear programming model. Section 3 describes the proposed algorithm for solving the problem. In order to assess the effect of the different criteria in our model and their impact on the solutions, a series of instances have been generated and solved. In Section 4 we describe this process and the obtained results. Furthermore, we illustrate the step by step performance of the solution algorithm with one of the considered benchmark instances. Finally, Section 5 offers concluding remarks.

2 Problem Statement and Model

An Integer Linear Programming (ILP) model is proposed where the objective function to be minimized is a dispersion measure, consisting of the sum of the distances to and from each edge to its assigned depot. This objective is equivalent to maximizing compactness.

Workload balance is modeled by setting upper and lower limits on the expected average total workload per district. Contiguity is obtained by assuring a path of allocated edges to a depot exists if an edge is to be assigned. Parity is enhanced by setting an upper limit on the total number of new odd degree nodes in each district allowed in the partition.

The underlying road network is modeled by an undirected planar graph $G = (V, E)$, where each edge of this graph corresponds to a road or street of the underlying road network. The node set V corresponds to street crossings or dead ends. We assumed G to be connected. Every edge $e = (i, j) \in E$ has a length, l_e, and a demand, d_e, which is assumed to be proportional to its length. Let $P \subset V$ denote a given subset of k depots. For simplicity, and without loss of generality, it is assumed that the k depots are labeled as nodes $1, 2, \ldots, k$. Therefore, in the following the district associated with depot $p \in P$ is referred to as district p. The problem is then defined as finding a valid k-partition of edges $E = (E_1, \ldots, E_k)$ such that for each $p \in P$ the district $G_p = (V(E_p), E_p)$ meets some required planning criteria. Here $V(E_p)$ represents the set of nodes that are incident to at least one edge of E_p.

Let $\sigma(e)$ denote the set of edges adjacent to edge $e \in E$ and $\delta(i)$ the set of edges incident to node $i \in V$. For any subset $S \subset E$, $\sigma(S)$ is the cut set of S, that is, the set of edges with one end point in $V(S)$, the set of nodes associated to edges in S, and the other end point in $V \setminus V(S)$.

The following parameters are known: b_{pe}, minimum distance from depot $p \in P$ to edge $e = (i, j) \in E$, defined as $\min\{f_{pi}, f_{pj}\}$, where f_{ij} is the shortest-path distance in G between nodes $i, j \in V$ relative to the length vector l; $\bar{D} = \sum_{e \in E} d_e/|P|$, average demand per district; $\tau_1 \in (0, 1)$, tolerance for demand balance constraints; $\tau_2 \in (0, 1)$, tolerance for parity constraints; and M, a sufficiently large number.

The following integer decision variables are defined:

x_{pe}, binary variable equal to 1 if edge $e \in E$ is assigned to depot $p \in P$ and 0 otherwise;

w_{pi}, binary variable equal to 1 if node $i \in V$ is incident to an edge assigned to depot $p \in P$ and 0 otherwise.

In addition, to model the parity of the vertices we use the following sets of variables: z^0_{ip}, binary variable equal to 1 if degree of node $i \in V$ in district $p \in P$ is odd and 0 otherwise; the auxiliary variables z_{ip}, which relate z^0_{ip} to the degree of node i, $i \in V$, $p \in P$; and r_i, binary variable equal to 1 if node i *loses* parity and 0 otherwise. Let $V^e \subset V$ be the set of even degree nodes in $G(V, E)$. A node $i \in V^e$ is said *to lose parity* if there is at least one district involving i where the degree of node i in that district is odd. In other words, node

i is said to keep its parity if the degree of i in each of its associated districts is even. In a similar fashion, let V^o be the set of odd degree nodes in $G(V, E)$. A node $i \in V^e$ is said *to lose parity* if there are at least two districts involving i where the degree of node i in those districts is odd. In other words, node i is said to keep its parity if the degree of i in each but one of its associated districts is even. Then, the proposed ILP is:

$$\text{Min} \quad g(x) = \sum_{p \in P} \sum_{e \in E} b_{pe} x_{pe} \tag{1}$$

$$\text{subject to} \quad \sum_{p \in P} x_{pe} = 1 \qquad e \in E \tag{2}$$

$$\sum_{s \in \sigma(S)} x_{ps} - \sum_{s \in S} x_{ps} \geq x_{pe} - |S| \qquad p \in P, e \in E, S \subset E \setminus \sigma(e) \tag{3}$$

$$\sum_{e \in E} d_e x_{pe} \leq \bar{D}(1 + \tau_1) \qquad p \in P \tag{4}$$

$$\sum_{e \in E} d_e x_{pe} \geq \bar{D}(1 - \tau_1) \qquad p \in P \tag{5}$$

$$\sum_{e \in \delta(i)} x_{pe} \leq M w_{pi} \qquad p \in P, i \in V \tag{6}$$

$$w_{pi} \leq \sum_{e \in \delta(i)} x_{pe} \qquad p \in P, i \in V \tag{7}$$

$$\sum_{e \in \delta(i)} x_{pe} = 2z_{ip} + z^0_{ip} \qquad p \in P, i \in V \tag{8}$$

$$r_i \leq \sum_{p \in P} z^0_{ip} \qquad i \in V^e \tag{9}$$

$$|P| r_i \geq \sum_{p \in P} z^0_{ip} \qquad i \in V^e \tag{10}$$

$$r_i \leq \sum_{p \in P} z^0_{ip} - 1 \qquad i \in V^o \tag{11}$$

$$|P| r_i \geq \sum_{p \in P} z^0_{ip} - 1 \qquad i \in V^o \tag{12}$$

$$\frac{1}{|V|} \sum_{i \in V} r_i \leq \tau_2 \tag{13}$$

$$w_{pi}, x_{pe}, z^0_{ip}, r_i \in \{0, 1\} \qquad p \in P, i, j \in V, e \in E \tag{14}$$

$$z_{ip} \in \mathbb{N} \cup \{0\} \qquad p \in P, i \in V \tag{15}$$

The objective function (1) manages the compactness of the district by measuring disper-

sion as the sum of distances to and from each edge to its allocated depot. It can be seen as a p-median type of function with the particularity that in our case the location of each depot is given and not part of the decision process. Constraints (2) force each edge to be assigned to exactly one depot. Constraints (3) ensure district connectivity. These constraints, analogous to the connectivity constraints used in [20] for node territory design, can be explained as follows. Let $S \subset E \setminus \sigma(e)$ be a subset whose edges are not adjacent to edge e in district p. If edge e is not assigned to p ($x_{pe} = 0$), the constraint becomes redundant. Furthermore, if there is at least one edge $s \in S$ that is not assigned to district p, then the second term of the left-hand side becomes strictly less than $|S|$ and the constraint becomes redundant too. Hence, constraints (3) become non-redundant only when all edges in S are assigned to district p. Then, the first term of the left-hand side must be greater than or equal to 1. That is, at least one edge in the cut set of set S must be assigned to district p as well. Applying the same rationale to set $S \cup \{s\}$ results in a territory connected to edge e in district p. Note that there is an exponential number of such constraints. Constraints (4) and (5) give rise to balanced districts within the allowed tolerance $\tau_1 \in (0, 1)$. Constraints (6)-(7) identify nodes involved in each district p, where M is a sufficiently large number. Typically M is given the value of largest node degree in the graph. In essence, they ensure that an edge is assigned to a depot if and only of its two incident nodes are assigned to the same depot, including depots. These constraints along with the connectivity constraints (3) imply that a depot p belongs district p; however, if this condition were to be relaxed the term $i \in V$ would have to be replaced by $i \in V \setminus P$ in constraints (6)-(7). Constraints (8) set the degree of node i in district p, and assign an appropriate value to z^0_{ip} in order to identify nodes of odd degree as well. Constraints (9)-(13), used to limit the imparity gain, deserve further explanation. First, for each node i of even degree in $G = (V, E)$, we can see that $r_i = 0$ if and only if $\sum_{p \in P} z^0_{ip} = 0$, that is, in each district involving node i its degree must be even for the node to be considered as keeping its parity ($r_i = 0$). This is achieved by constraints (9)-(10). Similarly, constraints (11)-(12) set the relationship for the imparity gain for each odd degree node. Finally, the percentage of nodes with lost imparity, given by $\sum_{i \in V} r_i$, is limited by parameter $\tau_2 \in (0, 1)$, in constraint(13). Finally, constraints (14) and (15) set the nature of the decision variables.

Note that constraints (9)-(13) are valid in any type of graph regardless its distribution of even and odd degree nodes; however, if the number of odd degree nodes in $G = (V, E)$ is

relatively large, constraints (9)-(13) may be replaced by (16)-(17) below

$$\sum_{i \in V} \sum_{p \in P} z_{ip}^{0} - l = l_0 \tag{16}$$

$$l \leq \tau_2 (l_0 + l) \tag{17}$$

where l is an integer variable that accounts for the imparity gain with the induced partition and parameter l_0 is the number of odd degree vertices in G. These new parity constraints reduce the number of binary variables in the model since the r_i variables are no longer needed. Computational experiments are done with both sets of parity constraints.

3 Solution Algorithm

The main difficulty for solving the proposed ILP arises from the exponential number of connectivity constraints; their explicit enumeration is practically impossible. Salazar-Aguilar et al. [21] deal with a similar issue but in the context of node-based territory design. To address this, they propose a solution algorithm that iteratively employs branch and bound and cut generation. In our work, we implement a similar idea to deal with the exponential number of connectivity constraints. To face this issue with the connectivity constraints, a solution algorithm as in Algorithm 1 in Salazar-Aguilar et al. [21] is implemented.

The idea is fairly simple. In our implementation, a relaxed ILP model, which does not include the connectivity constraints (3), is solved by branch and bound. The obtained solution is checked for disconnected districts. If disconnected districts are found, cuts are generated for violated connectivity constraints and added to the relaxed model. The relaxed model with such cuts is solved again and this is repeated until a solution with no violations is reported.

An alternative to this approach would be to replace the exponential number of connectivity constraints by a polynomial number of flow-based connectivity constraints [7]. However, for node districting problems, this approach has shown very little value as the size of the problem becomes very large. In addition, due to the structure of our problem, for practically all instances tested, the proposed algorithm converges in a single iteration. It was observed that solutions with disconnected districts were obtained on instances where the all or most of the depots are extremely close to each other. In districting applications (e.g., snow removal and disposal [11], salt spreading operations [15]) it is not typical to find situations where all or most of the depots or facilities are close to each other. In fact, in practically all of the districting literature where the locations of the depots or facilities are part of the

decision process, a compactness criterion, which makes the location of centers be dispersed, is considered.

Solution Algorithm 1

Step 1 Solve the ILP given by (1)-(15), with the connectivity constraints (3) relaxed.

Step 2 Identify if there are any disconnected districts, which would mean violated connectivity constraints are present. This reduces to identifying the connected components induced by the solution associated with each district. It is well known that finding the connected components in every district can be efficiently done in polynomial time by breadth first search (BFS).

Step 3 If violated constraints are found, generate the associated cuts, add them to the relaxed model and return to Step 1.

Step 4 If no constraints are violated, stop and return optimal solution.

The convergence of the algorithm is guaranteed due to two facts. First, the BFS algorithm returns either a set of violated connectivity constraints, in the form of unconnected subsets or an empty set. Second, there is a finite number of connectivity constraints. Thus the algorithm is guaranteed to stop at an optimal solution for the original model. The number of iterations the algorithm needs in practice to find an optimal solution is an issue to be investigated through experimental work.

4 Computational Experiments

The proposed model and solution algorithm was implemented within the GAMS framework (version 24.7) and solved using CPLEX 12.6 in an HP ProBook 6460b, Intel Core i5-2410M and CPU 2.3 GHz. For our experiments we used fifteen road networks from Belenguer et al. [1]. The authors classified their road networks in three groups, namely LPR-A, LPR-B, and LPR-C. We took five instances from each group (labeled LPR-A-01 to LPR-A-05, and so on). The size of the road network is the main difference among them. Now, since these instances, which represent road networks, are used by the authors for addressing mixed capacity arc routing problems (MCARPs) we have modified them to fit in our problem as follows. That database consists of mixed directed graphs with demand and service and transit costs associated to arcs. We take the same road topology ignoring the arc direction to form

an undirected graph. We use the distance between nodes as edge length. In addition, since we have depots as part of the problem parameters, we have extended this database of instances by duplicating some of these instances with different number of depots. In total, we have 20 base instances. Table 1 shows the properties and size of each of the instances in the database, where each row displays the instance name, number of nodes, number of edges, and number of integer variables (NIV) in the model.

Now, from these base dataset we have run many experiments varying some parameters such as the value of balance tolerance and imparity tolerance to address the important issues. Road networks with up to 401 nodes, 764 edges and 6 depots, resulting in 12,203 discrete variables were solved successfully.

Table 1: Data instances

Instance	$\|V\|$	$\|E\|$	NIV
LPR-A-01-2-*	28	52	300
LPR-A-01-3-*	28	52	436
LPR-A-02-2-*	53	92	553
LPR-A-03-2-*	146	252	1526
LPR-A-03-3-*	146	252	2216
LPR-A-03-4-*	146	252	2906
LPR-A-03-5-*	146	252	3596
LPR-A-04-4-*	195	339	3891
LPR-A-05-4-*	321	559	6409
LPR-B-01-2-*	28	47	290
LPR-B-01-3-*	28	47	421
LPR-B-02-2-*	53	90	551
LPR-B-03-4-*	163	292	3287
LPR-B-04-4-*	248	465	6293
LPR-B-05-6-*	401	764	12203
LPR-C-01-3-*	28	47	421
LPR-C-02-3-*	53	91	803
LPR-C-03-4-*	163	308	3351
LPR-C-04-3-*	277	514	4312
LPR-C-05-4-*	369	693	7569

Each instance of the extended data set and corresponding experiment is identified by the following notation. Instance LPR-A-01-2-10-05, for example, refers to network LPR-A-01 from Belenguer et al. [1], where the suffix "-2-10-05" means $p = 2$, and run under $\tau_1 = 0.10$ and $\tau_2 = 0.05$. The computation time limit was set to 60,000 seconds. When the algorithm stops due to this limit, the best feasible solution is shown. The relative optimality gap in the solver is set to 0.001%.

In total, when considering all different parameter variations a total of 89 different runs

on these 20 instances were attempted and successfully solved. Individual results for each run are displayed in the appendix. An important observation is that 86 out of these 89 runs needed just a single iteration to converge. This means the optimal or feasible solution found in the relaxed model has no disconnected components and therefore is optimal or feasible to the original model. This happens because the depots are somewhat dispersed throughout the road network and the dispersion minimization in the objective function favors connectivity around the depots. The three instances that needed more than a single iteration were built with the aim of getting disconnected districts in order to show the solution algorithm at work.

Now, among all these different runs we choose arbitrarily a subset with different values of p, τ_1, and τ_2, trying to cover many different combinations to test the model containing the alternative parity constraints. Optimal solutions were obtained to 50 out of 53 tested runs. The detailed results for each run are shown in the appendix. Here we summarize the most relevant results.

4.1 Effect of the Parity Constraints

In order to get a good estimation about the benefit of introducing the parity constraints in the model, a Chinese Postman Problem (CPP) was solved for each district once an optimal solution is found. This is done to get a measure of the deadhead a single vehicle would have to travel in each district. Our hypothesis is that introducing the parity constraints will lead to better districts for routing vehicles. This happens because there would be less deadhead compared to a model that does not include them and was proven to be true throughout all solved instances. Table 2 displays the results when solving a few instances with different values of τ_2 the parity tolerance parameter. Each row shows the instance name and the valued of the corresponding deadhead when solving a Chinese Postman Problem (CPP) for each district as a function of τ_2. Note that the $\tau_2 = 1.0$ column represents the situation when no parity constraints are present. The last column is relative improvement of the model when $\tau_2 = 0.01$ with respect to no using the parity constraints ($\tau_2 = 1.0$). The last row show the average relative improvements with respect to column ($\tau_2 = 1.0$) over all instances. For example, the value of 7.99 in the last row indicates that the average relative improvement of the deadhead when obtained under $\tau_2 = 1.0$ with respect to the absence of these constraints ($\tau_2 = 1.0$). As we can see, the introduction of the parity constraint yields lower values of nodes with lost parity which in turns causes a reduction in the total deadhead distance. The relative improvement shows values of 3.9-24.7 % when compared to the model where these constraints are absent. The average total improvement over all instances is 11.13 %. This is

indeed a positive impact.

Table 2: Effect of parity constraints in deadhead distances

Instance	$\tau_2 = 1.0$	0.10	0.04	0.03	0.02	0.01	RI (%)
LPR-A-03-3-*	7222.31	7222.31	6393.14	6393.14	6393.14	6143.33	14.9
LPR-A-04-4-*	9560.46	9560.46	9640.95	9199.70	8978.24	8584.50	10.2
LPR-A-05-4-*	15261.36	15261.36	15073.37	14872.61	14751.84	14255.61	6.6
LPR-B-01-2-*	1416.61	1416.61	1416.61	1259.78	1259.78	1259.78	11.1
LPR-B-03-4-*	8297.09	8297.09	8186.11	8143.17	8253.71	7895.36	4.8
LPR-B-04-4-*	10120.24	10120.24	9733.93	9123.67	8622.09	8282.4	18.2
LPR-B-05-6-*	16270.15	16270.15	16281.55	16281.55	15629.88	15005.6	7.8
LPR-C-01-3-*	1966.98	1528.21	2090.34	1681.27	1681.27	1681.27	14.5
LPR-C-02-3-*	4523.47	4461.14	3956.02	3838.51	3838.51	3407.74	24.7
LPR-C-03-4-*	8354.04	8354.04	8282.36	8089.14	8023.91	7419.36	11.2
LPR-C-04-3-*	11323.84	11323.84	11437.87	10996.33	10497.89	10670.64	5.8
LPR-C-05-4-*	15655.47	15655.47	15655.47	15830.33	15073.15	15044.66	3.9
Average RI (%)		1.97	1.92	6.26	7.99	11.13	

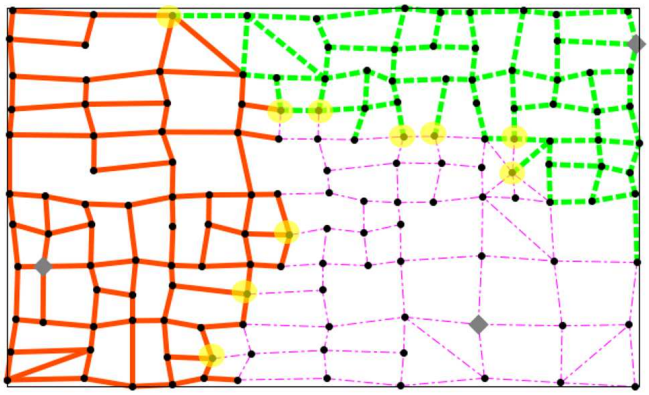
Of course, tightening the value of τ_2 must also have a negative effect in the objective function. Table 3 displays similar results as the previous table showing the value of the objective function for the same set of instances. The last column shows the relative increment of the objective function when $\tau_2 = 0.01$ is in effect with respect to no use of the parity constraints ($\tau_2 = 1.0$). As we can see, the detriment of the objective function is very marginal for practically all instances. The worst case was instance LPR-C-01-02 that observed a 3.18 % increase. The average over all instances was 0.56 %.

Table 3: Effect of parity constraints in objective function values

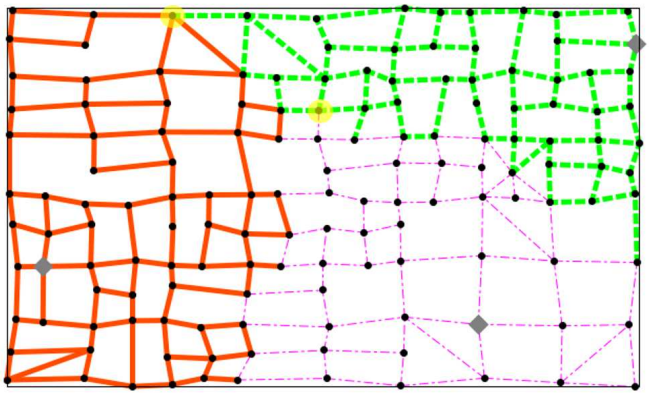
Instance	$\tau_2 = 1.0$	0.10	0.04	0.03	0.02	0.01	RI (%)
LPR-A-03-3-*	277194.12	277194.12	277741.11	277741.11	277741.11	278068.71	0.31
LPR-A-04-4-*	291013.88	291013.88	291132.37	291255.70	291484.67	291746.50	0.25
LPR-A-05-4-*	710050.53	710050.53	710116.40	710218.53	710478.15	710921.49	0.12
LPR-B-01-2-*	28318.27	28318.27	28318.27	28527.56	28527.56	28527.56	0.73
LPR-B-03-4-*	249074.20	249074.20	249090.10	249119.28	249150.66	249227.11	0.06
LPR-B-04-4-*	380014.97	380014.97	380076.63	380221.68	380536.59	380785.88	0.20
LPR-B-05-6-*	711108.32	711108.32	711122.88	711122.88	711699.23	711990.07	0.12
LPR-C-01-3-*	17325.62	17501.45	17635.76	17894.65	17894.65	17894.65	3.18
LPR-C-02-3-*	58836.93	58847.73	59021.54	59267.68	59267.68	59583.99	1.25
LPR-C-03-4-*	263035.3	263035.3	263171.69	263321.96	263404.52	263760.4	0.27
LPR-C-04-3-*	943293.89	943293.89	943318.28	943425.82	943619.75	944016.38	0.08
LPR-C-05-4-*	853979.50	853979.50	853979.50	854018.26	854197.796	854623.55	0.08

Let us now see some graphical results. Consider the two results for example LPR-A-03 in Figure 3. This is a graph with 146 nodes, 252 edges, 3 depots whose corresponding model

has 2216 discrete variables. The motivation for these instances is to show how the parity restriction works and affects the optimal solution in a graphical way.



(a) Parity tolerance = 20%



(b) Parity tolerance = 5%

Figure 3: Effect of parity tolerance on instances LPR-A-03-3-20

The network in Figure 3(a) used $\tau_2 = 0.20$, which is equivalent, in this particular case, to relaxing the parity constraints. The network in Figure 3(b) used $\tau_2 = 0.05$. After solving each instance, the network on Figure 3(a) has $\sum_{i \in V} r_i = 10$, which means 10 nodes lost parity, that is, either they had even degree in the original graph and have now odd degree, or they were of odd degree in the original graph and are of odd degree in more than one district due to the partition now. Such nodes are encircled in the figure. Its corresponding deadhead distance is 7222.31. In contrast, the solution on Figure 3(b) has only 2 nodes that lost parity. Its corresponding deadhead distance is 6393.14. This represents a 11.4 % improvement. This is exactly what the model is trying to convey.

Care must be taken when setting the values of τ_1 and τ_2 since they are related to each other and wrong combinations of the pair will lead to empty feasible regions. The parity tolerance τ_2 has a lower bound threshold value determined by the balance tolerance; beneath

this value the problem becomes infeasible. If the balance tolerance is completely relaxed (τ_1=1), parity tolerance τ_2 can be as low as zero. This means that if no better partition was found, all arcs could be assigned to a single district where no imparity would be gained. The balance tolerance, τ_1, on the other side, has a lower bound threshold value that depends not only on the imparity tolerance but also on specific characteristics of each instance such as the number of districts, the number of edges, and the demand for each edge.

4.2 Effect of Balance Constraints

The effect of moving the balance tolerance is now studied. Figure 4 shows results for network LPR-B-02-2 with ten different values of τ_1. The balance constraint parameter varies from high (left-most) to low (right-most) in the figure. Parameter τ_1 starts at a value of 20%, which is equivalent to having the balance constraints relaxed and decreases to the value of zero. Not all instances allow having zero tolerance balance, in fact most would be infeasible, but this example can produce perfectly balanced districts.

The objective function increases as the value of τ_1 is tightened as expected, but no correlation was found in solution time. As far as imparity is concerned, an increase, although not monotone, was observed.

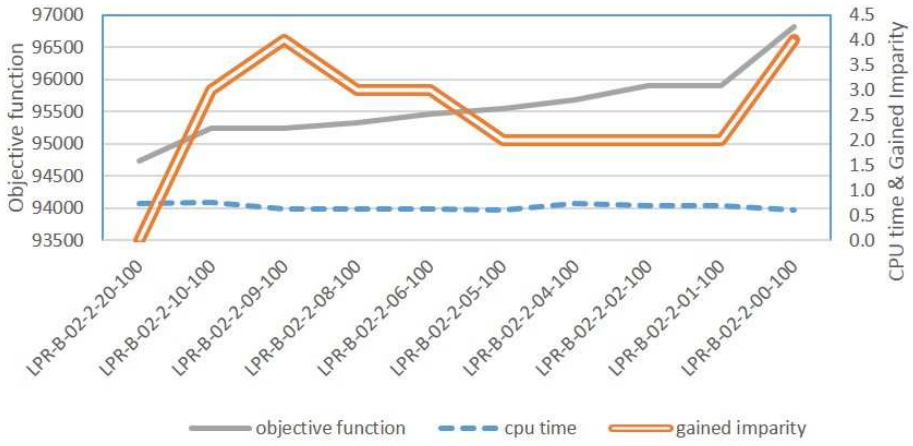


Figure 4: Effect of balance constraints

4.3 Effect of Number and Location of Depots

We now investigate the effect of the number and location of depots on the computational effort. To examine this effect a collection of different instances were created from network

LPR-A-03. This network has 146 nodes and 252 edges. Instance LPR-A-03-2, has two depots, instance LPR-A-03-3, has three depots. Instances LPR-A-03-4, LPR-A-03-4a, LPR-A-03-4b, LPR-A-03-4c and LPR-A-03-4d all have 4 differently located depots. Finally instance LPR-A-03-5 has 5 depots.

Figure 5 plots the CPU time employed versus the number of discrete variables in every instance. It can be seen how for a specific network the number of binary variables increases linearly with the number of depots, and this is related to the time required to solve if a solution is reached in one iteration. This suggests that the position of the depots is of great importance as it directly affects computation time. It was observed that instances where the depots are more dispersed tend to be solved in a single iteration of the algorithm.

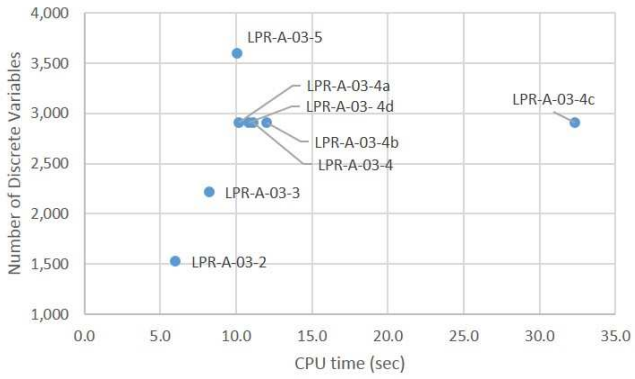


Figure 5: Computation times and model size for instance LPR-A-03

.

In Figure 5 it can be seen the solution time for instance LPR-A-03-4c is three times larger than for the rest of the instances with four depots. This is so because the algorithm had to run three iterations in order to find a connected solution. It is the hardest instance to solve for LPR-A-03.

Throughout the experimentation, we have focused our work on instances where the depots are relatively dispersed in the network, which is more representative of real-world instances. We have seen how, for these type of instances, the solution algorithm converges in a single iteration. However, there might be other real-world cases where the depots are not necessarily dispersed. Thus, we are also interested in investigating the performance of the model on these type of instances.

A series of additional instances were then created in an atypical manner, with all or some of the depots placed close to each other. This was done in order to force the relaxed model to produce partitions with disconnected districts, so that the complete algorithm can be shown

at work, and observe the model behavior under these circumstances. Examples LPR-A-03-4c and Example LPR-A-03-4d are two of such instances.

Partitions for all instances of LPR-A-03 with four depots are seen in Figure 6. LPR-A-03-4a has the depots located on the outskirts of the network and LPR-A-03-4b has the depots ending up at the center of their district. LPR-A-03-4c has three aligned depots neighboring each other and finally LPR-A-03-4d has two pairs of neighboring depots. Figure 6 shows that having close depots make the districts more dispersed. This happens because the model tries to build each district around the depot, if two depots are close together, they compete with each other and dispersion results.

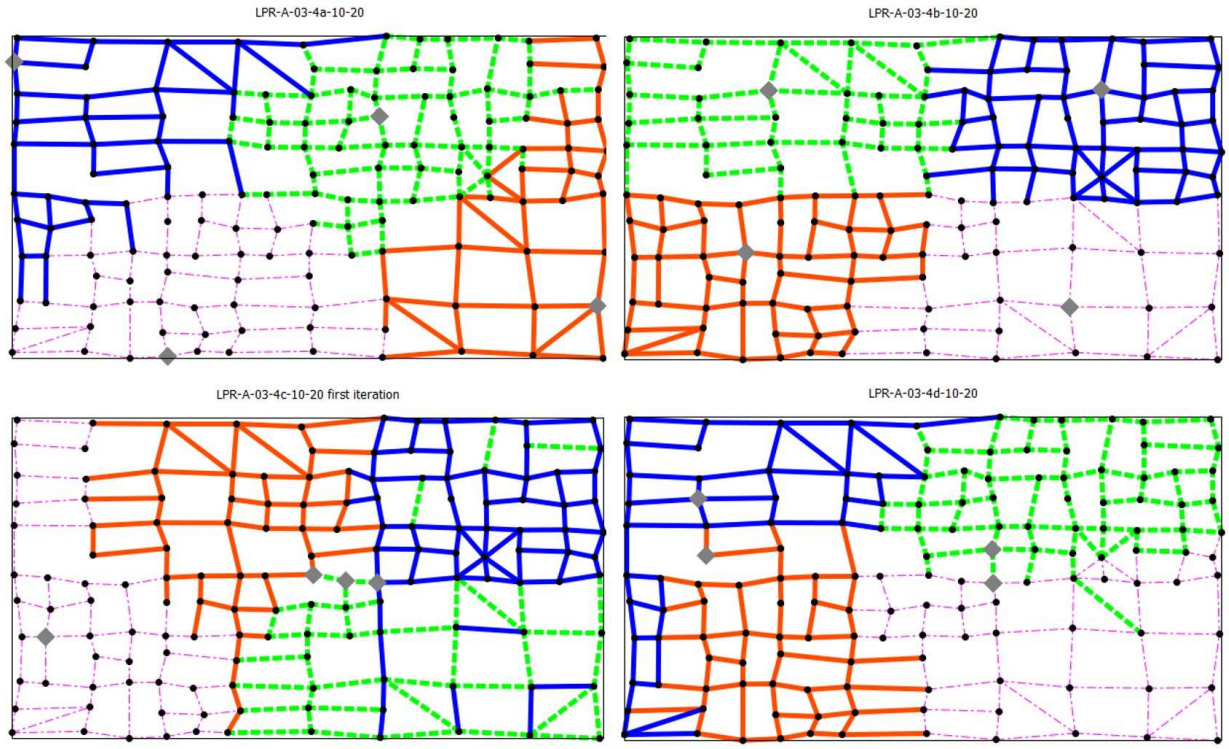


Figure 6: Results for LPR-A-03

.

Note how partition for LPR-A-03-4c is disconnected and therefore is infeasible to the original model. Although LPR-A-03-4d has neighboring depots also, the optimal solution was found in a single iteration. For LPR-A-03-4c cuts were generated and the model was solved again until no disconnected components were found in the solution. For this particular instance, we illustrate in the following subsection how the algorithm works to find a feasible solution.

4.4 Illustrative Example

We now present an example that illustrates every step of the algorithm. To this end, we use instance LPR-A-03-4c which has three depots close together to force disconnected districts. Figure 7 presents the result for the first iteration. To better visualize the algorithmic process, the edge label is displayed in some edges. The depots are shown in gray diamonds. The solution found in each iteration can be seen in Figures 8 and 9, including the optimal solution where the districts are all connected.

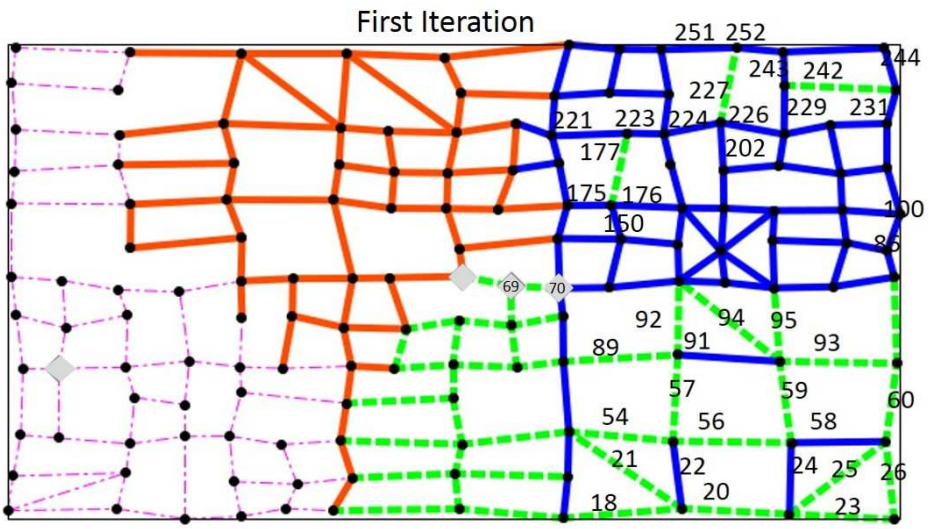


Figure 7: Iteration 1 for LPR-A-03-4c-10-20

.

Solution Algorithm for instance LPR-A-03-4c-10-20.

1. Solve the model with connectivity constraints (3) relaxed.
   - Solver solution: 243972.95
   - CPU time (sec.): 7.083
   - The resulting partition is displayed in Figure 7.
2. Solve the separation problem by identifying districts with disconnected components.
   - In this example two districts are disconnected, the two on the right. The lower district whose corresponding depot is at node 69, and the upper right district which has its depot on node 70.

3. For each disconnected district, generate the appropriate cuts and add them to the relaxed model. In the previous step two disconnected districts were identified with three disconnected segments each.

   - The lower district (depot at node 69) has the following disconnected subsets of edges: $S_1 = \{177\}$, $S_2 = \{227\}$, and $S_3 = \{242\}$, with corresponding cut sets $\sigma(S_1) = \{150, 175, 176, 221, 223\}$, $\sigma(S_2) = \{202, 224, 226, 251, 252\}$, and $\sigma(S_3) = \{229, 231, 243, 244\}$, respectively. These will generate the following connectivity cuts (3):

$$x_{69,150} + x_{69,175} + x_{69,176} + x_{69,221} + x_{69,223} - x_{69,177} \geq 0$$
$$x_{69,202} + x_{69,224} + x_{69,226} + x_{69,251} + x_{69,252} - x_{69,227} \geq 0$$
$$x_{69,229} + x_{69,231} + x_{69,243} + x_{69,244} - x_{69,242} \geq 0$$

   - The upper right district (depot at node 70) has the following disconnected subsets of edges: $S_1 = \{22\}$, $S_2 = \{24, 58\}$, and $S_3 = \{91\}$, with corresponding cut sets $\sigma(S_1) = \{18, 20, 21, 54, 56, 57\}$, $\sigma(S_2) = \{20, 23, 26, 56, 59, 60\}$, and $\sigma(S_3) = \{57, 59, 89, 92, 93, 94, 95\}$, respectively. These will generate the following cuts:

$$x_{70,18} + x_{70,20} + x_{70,21} + x_{70,54} + x_{70,56} + x_{70,57} - x_{70,22} \geq 0$$
$$x_{70,20} + x_{70,23} + x_{70,26} + x_{70,56} + x_{70,59} + x_{70,60} - (x_{70,24} + x_{70,58}) \geq -1$$
$$x_{70,57} + x_{70,59} + x_{70,89} + x_{70,92} + x_{70,93} + x_{70,94} + x_{70,95} - x_{70,91} \geq 0$$

4. Iteration 2: Solve the relaxed model with added constraints.

   - Solver solution: 244045.07
   - CPU time (sec.): 12.449
   - The resulting partition is displayed in Figure 8.

5. Solve separation problem by identifying districts with disconnected components.

   - In this iteration the same districts (associated to depots 69 and 70) are found disconnected.

6. For each disconnected district, generate the appropriate cuts and add them to the relaxed model.

   - As can be seen in Figure 8, district 69 has one disconnected component and district

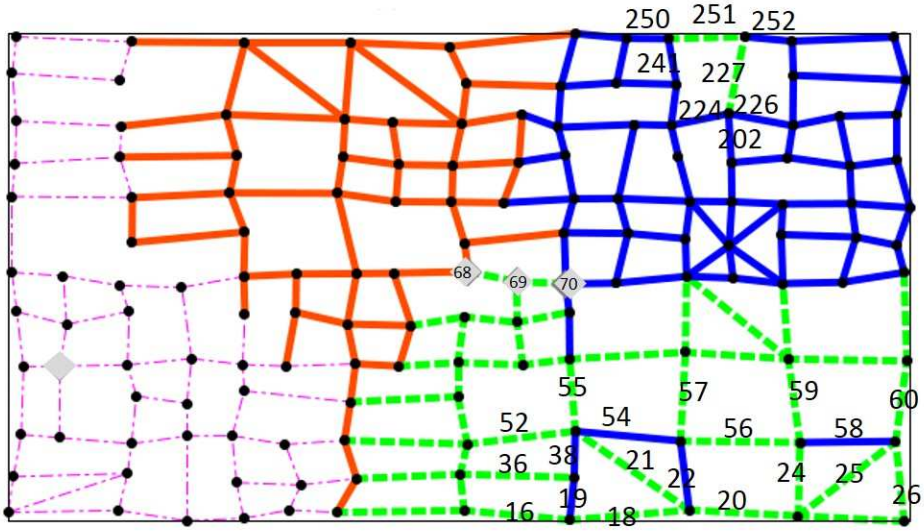


Figure 8: Iteration 2 for LPR-A-03-3

70 has two disconnected components.

- The lower right district (depot at node 69) has one disconnected component $S_1 = \{227, 251\}$ with corresponding cut set $\sigma(S_1) = \{202, 224, 226, 241, 250, 252\}$. This generates the following cut:

$$x_{69,202} + x_{69,224} + x_{69,226} + x_{69,241} + x_{69,250} + x_{69,252} - (x_{69,227} + x_{69,251}) \geq -1$$

- The upper right district (depot at node 70) has two disconnected components $S_1 = \{58\}$ and $S_2 = \{19, 22, 38, 54\}$, with corresponding cut sets $\sigma(S_1) = \{24, 25, 26, 56, 59, 60\}$ and $\sigma(S_2) = \{16, 20, 36, 52, 55, 56, 57\}$, respectively. These generate the following cuts:

$$x_{70,24} + x_{70,25} + x_{70,26} + x_{70,56} + x_{70,59} + x_{70,60} - x_{70,58} \geq -0$$

$$x_{70,16} + x_{70,20} + x_{70,36} + x_{70,52} + x_{70,55} + x_{70,56} + x_{70,57} - (x_{70,19} + x_{70,22} + x_{70,38} + x_{70,54}) \geq -3$$

7. Iteration 3: Solve relaxed model with added constraints.

- Solver solution: 244045.07
- CPU time (sec.): 12.792

8. Solve separation problem by identifying districts with disconnected components.

    - No districts are disconnected

9. Solution is optimal.

    - Solver solution: 244045.07
    - TOTAL CPU time (sec.): 32.32
    - Optimal solution partition is displayed in Figure 9.

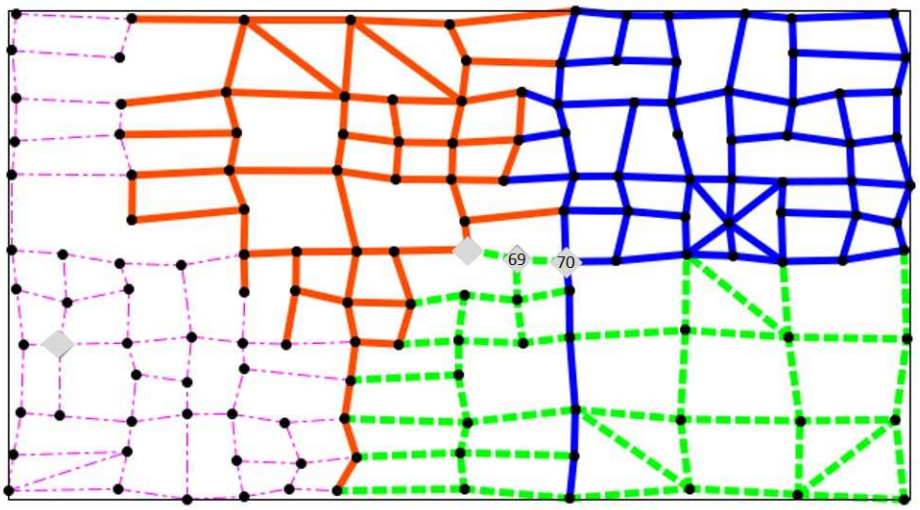


Figure 9: Iteration 3 for LPR-A-03-3

4.5 Assesment of Alternative Parity Constraints

Two different sets of parity constraints are proposed for modeling districts for routing services. In the previous section, the empirical work was based on the model with parity constraints (9)-(13). As pointed out earlier, these constraints are always valid for any type of graph. However, there is a class of instances where parity constraints (16)-(17) may be used instead. The advantage is that constraints (16)-(17) are smaller in size and seem a good option. However, the main disadvantage is that these may fail when the number of odd degree nodes is relatively low (that is when the graph tends to be Eulerian).

In most of the cases, solving the model with either set of parity constraints rendered the same optimal solution, although instances solved with constraints (9)-(13) found a solution faster. Figure 10 shows the instances solved by both sets of constraints, in almost all the

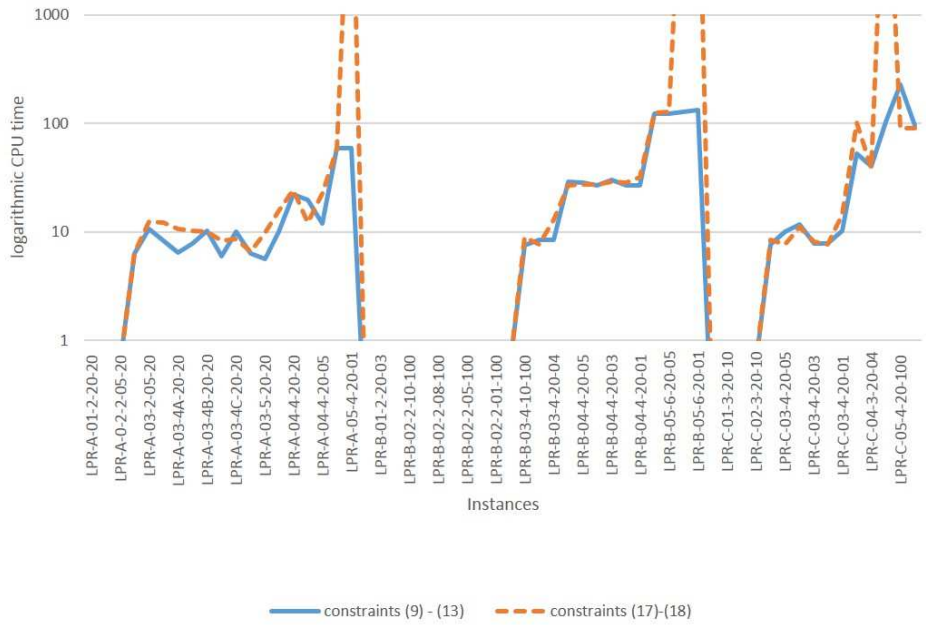


Figure 10: Comparison between parity constraints

.

cases using constraints (9)-(13) results in shorter computation times, even though the model has more discrete variables. The dashed line in the figure is always above the solid line of constraints (9)-(13). Logarithmic scale is used in Figure 10 since some computations take only a fraction of a second and some up to 60,000 seconds. Three instances, LPR-B-05-6-20-02, LPR-B-05-6-20-01 and LPR-A-05-4-20-01, were timed out at 60,000 seconds using constraints (16)-(17). They are the three dashed points that do not appear in the graph due to the scale. Constraints (9)-(13) solved these instances in around a hundred seconds, therefore they adequately favor parity in the model, and prove to be more efficient in solving all instances. Table 6 in the appendix contains all the results for the instances that were solved with constraints (9)-(13) replaced by constraints (16)-(17).

5 Closing Remarks

A districting model that services arc routing activities is proposed. The model produces partitions that are contiguous, have a work load balance, acquired imparity under a given tolerance, and have minimal dispersion around the depots.

The use of parity constraints is a novelty. Odd degree nodes in district partitions impact

vehicle routing since they translate into deadhead time. We introduce a set of constraints that limits imparity and could be helpful in any other districting models that lead to vehicle routing.

An exact solution procedure based on branch and bound with a cut generation strategy was successfully applied to solve the model to optimality. Despite the exponential number of connectivity constraints, the solution method can be easily implemented with any off-the-shelve branch-and-bound solver. The algorithm solves, in a given iteration, a model with the connectivity constraints relaxed. If an unconnected partition is produced, a valid inequality is generated and added and the model is resolved. This is repeated iteratively until a feasible and thus optimal solution is found. The algorithm was able to solve most of the instances tested in a single iteration. This is due in great extent to the disperse location of the depots. However, we provided some examples on how the method successfully solves less common instances with closer depots generating cuts and finding the solution in a few iterations. Compactness is favored on networks where the depots end up at the center of their district due to the objective function of the model. If a specific problem calls for neighboring or close depots another objective function would be recommended, for example, one that takes into account distance between edges instead of their distance to the depot.

Series of instances were solved to determine the impact of the parity constraint on the objective function and resulting partitions. Networks with up to 491 nodes, 763 edges whose associated model has 12,203 discrete variables were solved successfully. By having the parity constraints in the model, solutions with less deadhead distances are possible. This is of great impact to the routing of vehicles.

The model is useful at a tactical level as it can be used to promote characteristics of interest to specific applications. Parameters can be adjusted for workload balance, and parity which traduces to deadhead distance. This generality produces solutions with different characteristics, depending on how their tolerances are set. A sensitivity analysis between these features has been studied and some results presented. The obtained results indicate that the parity constraints seem useful as it leads to partitions that allow efficient vehicle routing. Future work in developing heuristics is suggested by the impossibility of finding feasible solutions for very large instances.

Possible extensions of this research worth pursuing involve the study of districting problems with districting-routing decisions. When both districting and routing decisions are to be taken into account simultaneously, the way the service/traversal of an arc becomes critical. For instance, depending on the application, it could happen that streets are not served in one

traversal or it is not allowed to traverse the streets in both directions. In some applications more than one tour could be assigned to each depot. Furthermore, there are applications where the demand is not necessarily proportional to the length of the street as assumed in our work. For example, for meter reading a small street with some apartment blocks may have a higher demand than a large street with some single-family houses.

In this work, we maximize the compactness and regard balance and deadheading distance as constraints. Hence, it would be very interesting to study the problem from a multi-objective programming approach and develop approximations to the Pareto-optimal front. For instance, an interesting question arises when assessing the deadhead distances found when optimizing compactness. That is, if one wanted to find the optimal deadhead distances and treat compactness as a constraint the resulting MILP model is harder to solve.

Acknowledgements: The paper has been improved thanks to the remarks of three anonymous referees. This research has been partially funded by Tecnologico de Monterrey Research Research Group in Industrial Engineering and Numerical Methods 0822B01006, by the Mexican National Council for Science and Technology (CONACyT) through grant SEP-CONACyT CB-2011-01-166397, by Universidad Autónoma de Nuevo León through grant UANL-PAICYT CE728-11, and by the Spanish Ministry of Economia y Competitividad and ERDF funds through grant MTM2012-36163-C06-05.

References

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

[2] L. Bodin and L. Levy. The arc partitioning problem. *European Journal of Operational Research*, 53(3):393–401, 1991.

[3] A. Butsch, J. Kalcsics, and G. Laporte. Districting for arc routing. *INFORMS Journal on Computing*, 26(4):809–824, 2014.

[4] J. F. Campbell and A. Langevin. Operations management for urban snow removal and disposal. *Transportation Research*, 29(5):359–370, 1995.

[5] J. F. Campbell and A. Langevin. The snow disposal assignment problem. *Journal of the Operational Research Society*, 46(8):919–929, 1995.

[6] J. C. Duque, R. Ramos, and J. Suriñach. Supervised regionalization methods: A survey. *International Regional Science Review*, 30(3):195–220, 2007.

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

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

[9] D. L. Horn, C. R. Hampton, and A. J. Vandenberg. Practical application of district compactness. *Political Geography*, 12(2):103–120, 1993.

[10] J. Kalcsics, S. Nickel, and M. Schröder. Towards a unified territorial design approach: Applications, algorithms, and GIS integration. *TOP*, 13(1):1–56, 2005.

[11] A. Labelle, A. Langevin, and J. F. Campbell. Sector design for snow removal and disposal in urban areas. *Socio-Economic Planning Sciences*, 36(3):183–200, 2002.

[12] H. Y. Lin and J. J. Kao. Subregion districting analysis for municipal solid waste collection privatization. *Journal of the Air and Waste Management Association*, 58(1):104–111, 2008.

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

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

[15] L. Muyldermans, D. Cattrysse, D. Van Oudheusden, and T. Lotan. Districting for salt spreading operations. *European Journal of Operational Research*, 139(3):521–532, 2002.

[16] N. Perrier, A. Langevin, and J. F. Campbell. A survey of models and algorithms for winter road maintenance. part i: System design for spreading and plowing. *Computers & Operations Research*, 33(1):209–238, 2006.

[17] N. Perrier, A. Langevin, and J. F. Campbell. A survey of models and algorithms for winter road maintenance. part ii: System design for snow disposal. *Computers & Operations Research*, 33(1):239–262, 2006.

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

[19] F. Ricca, A. Scozzari, and B. Simeone. Political districting: From classical models to recent approaches. *Annals of Operations Research*, 204(1):271–299, 2013.

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

[21] M. A. Salazar-Aguilar, R. Z. Ríos-Mercado, and M. Cabrera-Ríos. New models for commercial territory design. *Networks and Spatial Economics*, 11(3):487–507, 2011.

[22] L. Silva de Assis, P. M. Franca, and F. L. Usberti. A redistricting problem applied to meter reading in power distribution networks. *Computers & Operations Research*, 41:65–75, 2014.

[23] A. A. Zoltners and P. Sinha. Sales territory design: Thirty years of modeling and implementation. *Marketing Science*, 24(3):313–331, 2005.

Appendix

The results of each tested instance is shown in Table 4, Table 5, and Table 6. Table 4 contains the results of all instances solved by the model that reached optimal solution in one iteration. Table 6 contains instances that were solved with constraints (9)-(13) replaced by constraints (16)-(17). Both tables have nine columns. First three columns of the tables are related with the characteristics of the problem and state the number, the name, and the discrete variables of each of the instances. Next three columns are related with the objective function presenting the computation time in cpu seconds, the solver gap, if any, and the objective value. The next two columns show the gained imparity $\sum_{i \in V} r_i$ and the imparity quotient $\frac{1}{|V|} \sum_{i \in V} r_i$, which in the model is limited by τ_2. Finally the last column is the sum all districts of the calculated deadhead distance a single vehicle would drive in order to service a district.

Table 4: Results for all LPR instances solved with original model in one iteration

	Instance	NIV	Time	Gap	Objective	$\sum_{i \in V} r_i$	$\frac{1}{\lvert V \rvert} \sum_{i \in V} r_i$	Deadhead
1	LPR-A-01-2-20-20	300	0.047	0	28225.72	1	0.0357	1404.31
2	LPR-A-01-3a-20-20	436	0.265	0	19545.77	2	0.0714	1791.73
3	LPR-A-01-3b-20-20	436	0.125	0	24381.38	4	0.1429	1793.53
4	LPR-A-02-2-20-20	553	0.266	0	104262.04	2	0.0377	3382.83
5	LPR-A-02-2-10-20	553	0.265	0.0008	104279.75	2	0.0377	3535.82
6	LPR-A-02-2-05-20	553	0.140	0	104364.48	2	0.0377	3517.42
7	LPR-A-03-2-20-20	1526	0.203	0	338238.35	3	0.0205	6649.08
8	LPR-A-03-2-10-20	1526	0.655	0	338238.35	3	0.0205	6649.08
9	LPR-A-03-3-20-20	2216	0.156	0	277194.12	10	0.0685	7222.31
10	LPR-A-03-3-20-05	2216	0.432	0	277741.11	2	0.0137	6393.14
11	LPR-A-03-3-20-01	2216	0.515	0	278068.71	0	0.0000	6143.33
12	LPR-A-03-3-10-20	2216	0.655	0	278236.73	6	0.0411	7085.64
13	LPR-A-03-4-10-20	2,906	0.795	0	218692.54	6	0.0411	7949.95
14	LPR-A-03-4A-20-20	2906	0.296	0	225444.46	4	0.0274	7674.86
15	LPR-A-03-4A-10-20	2906	0.982	0	230674.05	5	0.0342	7632.82
16	LPR-A-03-4B-20-20	2906	0.390	0	205955.13	9	0.0616	7721.61
17	LPR-A-03-4B-10-20	2906	0.897	0	210081.05	10	0.0685	7942.34
18	LPR-A-03-4D-20-20	2906	0.624	0	234829.70	13	0.0890	7830.63
19	LPR-A-03-4D-10-20	2906	0.789	0.000009	235274.03	16	0.1096	8091.29
20	LPR-A-03-5-20-20	3596	0.374	0	182090.43	11	0.0753	7988.49
21	LPR-A-03-5-10-20	3596	0.733	0	184797.06	11	0.0753	7685.05
22	LPR-A-04-4-20-100	3891	0.748	0	291013.88	11	0.0564	9560.46
23	LPR-A-04-4-20-10	3891	0.748	0	291013.88	11	0.0564	9560.46
24	LPR-A-04-4-20-04	3891	0.858	0	291132.37	7	0.0360	9640.95
25	LPR-A-04-4-20-03	3891	1.357	0.000008	291255.70	5	0.0260	9199.70
26	LPR-A-04-4-20-02	3891	0.702	0.000004	291484.67	3	0.0150	8978.24
27	LPR-A-04-4-20-01	3891	0.874	0	291746.50	1	0.0050	8584.50
28	LPR-A-04-4-10-20	3891	0.390	0	294378.01	10	0.0513	9722.13
29	LPR-A-05-4-20-100	6409	0.889	0	710050.53	17	0.0530	15261.36
30	LPR-A-05-4-20-05	6409	0.593	0.000003	710071.45	14	0.0436	15105.16
31	LPR-A-05-4-20-04	6409	0.624	0.000009	710116.40	12	0.0374	15073.37
32	LPR-A-05-4-20-03	6409	0.624	0	710218.53	9	0.0280	14872.61
33	LPR-A-05-4-20-02	6409	1.435	0	710478.15	6	0.0187	14751.84
34	LPR-A-05-4-20-01	6409	0.718	0.000007	710921.49	3	0.0093	14255.61
35	LPR-B-01-2-20-100	290	0.109	0	28318.27	1	0.0357	1416.61
36	LPR-B-01-2-20-03	290	0.047	0	28527.56	0	0.0000	1259.78
37	LPR-B-02-2-20-100	551	0.234	0	94741.95	0	0.0000	3142.43
38	LPR-B-02-2-10-100	551	0.234	0	95237.12	3	0.0566	3918.10

	Instance	NIV	Time	Gap	Objective	$\sum_{i \in V} r_i$	$\frac{1}{\lvert V \rvert} \sum_{i \in V} r_i$	Deadhead
39	LPR-B-02-2-09-100	551	0.140	0	95238.63	4	0.0755	3663.00
40	LPR-B-02-2-08-100	551	0.203	0	95325.27	3	0.0566	3466.25
41	LPR-B-02-2-06-100	551	0.218	0	95458.07	3	0.0566	3486.76
42	LPR-B-02-2-05-100	551	0.094	0	95547.84	2	0.0377	3104.21
43	LPR-B-02-2-04-100	551	0.156	0	95680.64	2	0.0377	3124.72
44	LPR-B-02-2-02-100	551	0.198	0	95908.54	2	0.0377	3045.28
45	LPR-B-02-2-01-100	551	0.156	0	95908.54	2	0.0377	3045.28
46	LPR-B-02-2-00-100	551	104.100	0	96820.48	4	0.0755	3708.30
47	LPR-B-03-4-10-100	3287	0.234	0	249796.77	7	0.0429	8086.31
48	LPR-B-03-4-20-100	3287	0.234	0	249074.20	7	0.0429	8297.09
49	LPR-B-03-4-20-04	3287	0.301	0	249090.10	6	0.0368	8186.11
50	LPR-B-03-4-20-03	3287	0.359	0	249119.28	4	0.0245	8143.17
51	LPR-B-03-4-20-02	3287	0.328	0	249150.66	3	0.0184	8253.71
52	LPR-B-03-4-20-01	3287	0.374	0	249227.11	1	0.0061	7895.36
53	LPR-B-04-4-20-100	6293	29.141	0	380014.97	14	0.0565	10120.24
54	LPR-B-04-4-20-05	6293	28.282	0.000004	380016.52	12	0.0484	9888.36
55	LPR-B-04-4-20-04	6293	26.738	0	380076.63	9	0.0363	9733.93
56	LPR-B-04-4-20-03	6293	30.311	0	380221.68	7	0.0282	9123.67
57	LPR-B-04-4-20-02	6293	27.019	0	380536.59	4	0.0161	8622.09
58	LPR-B-04-4-20-01	6293	26.739	0	380785.88	2	0.0081	8282.40
59	LPR-B-05-6-20-100	12203	122.320	0.000009	711108.32	22	0.0549	16270.15
60	LPR-B-05-6-20-05	12203	123.506	0.000009	711122.88	20	0.0499	16281.55
61	LPR-B-05-6-20-02	12203	126.221	0.00001	711699.23	8	0.0200	15629.88
62	LPR-B-05-6-20-01	12203	133.365	0.000007	711990.07	4	0.0100	15005.60
63	LPR-C-01-3-20-100	421	0.218	0	17325.62	5	0.1786	1966.98
64	LPR-C-01-3-20-10	421	0.031	0	17501.45	2	0.0714	1528.21
65	LPR-C-01-3-20-04	421	0.249	0	17635.76	1	0.0360	2090.34
66	LPR-C-01-3-20-03	421	0.234	0	17894.65	0	0.0000	1681.27
67	LPR-C-02-3-20-100	803	0.609	0	58836.93	6	0.1132	4523.47
68	LPR-C-02-3-20-10	803	0.624	0	58847.73	5	0.0943	4461.14
69	LPR-C-02-3-20-04	803	0.452	0	59021.54	2	0.0380	3956.02
70	LPR-C-02-3-20-03	803	0.483	0	59267.68	1	0.0190	3838.51
71	LPR-C-02-3-20-01	803	0.483	0	59583.99	0	0.0000	3407.74
72	LPR-C-03-4-20-100	3351	7.582	0	263035.30	10	0.0614	8354.04
73	LPR-C-03-4-20-05	3351	10.031	0	263123.05	8	0.0491	8361.60
74	LPR-C-03-4-20-04	3351	11.638	0	263171.69	6	0.0368	8282.36
75	LPR-C-03-4-20-03	3351	7.754	0	263321.96	4	0.0245	8089.14
76	LPR-C-03-4-20-02	3351	7.847	0	263404.52	3	0.0184	8023.91
77	LPR-C-03-4-20-01	3351	10.140	0	263760.40	1	0.0061	7419.36
78	LPR-C-04-3-20-100	4312	52.494	0.000009	943293.89	13	0.0469	11323.84
79	LPR-C-04-3-20-04	4312	40.404	0	943318.28	11	0.0397	11437.87

	Instance	NIV	Time	Gap	Objective	$\sum_{i \in V} r_i$	$\frac{1}{\lvert V \rvert} \sum_{i \in V} r_i$	Deadhead
80	LPR-C-04-3-20-03	4312	102.914	0.00001	943425.82	8	0.0289	10996.33
81	LPR-C-04-3-20-02	4312	99.529	0.000009	943619.75	5	0.0181	10497.89
82	LPR-C-04-3-20-01	4312	102.103	0.000009	944016.38	2	0.0072	10670.64
83	LPR-C-05-4-20-100	7569	225.312	0.000005	853979.50	12	0.0325	15655.47
84	LPR-C-05-4-20-03	7569	94.552	0.000008	854018.26	11	0.0298	15830.33
85	LPR-C-05-4-20-02	7569	119.839	0.000008	854197.80	7	0.0190	15073.15
86	LPR-C-05-4-20-01	7569	180.743	0.000009	854623.55	3	0.0081	15044.66

Table 5: Instances that required more than one iteration

	Instance	NIV	Time	Gap	Objective	Iterations
87	LPR-A-03-4c-10-20	2906	32.324	0.000010	244045.1	3
88	LPR-A-01-3-20-20	436	4.380	0.000009	22276.2	36
89	LPR-B-01-3-20-20	421	2.602	0.000002	24187.8	22

Table 6: Results for model solved with constraints (9)-(13) replaced by constraints (16)-(17)

	Instance	NIV	Time	Gap	Objective	$\sum_{i \in V} r_i$	$\frac{1}{\lvert V \rvert} \sum_{i \in V} r_i$	Deadhead
90	LPR-A-01-2-20-20	272	0.11	0	28225.72	1	0.0357	1404.3
91	LPR-A-02-2-20-20	500	0.61	0	104262.04	2	0.0377	3382.8
92	LPR-A-02-2-05-20	500	0.59	0	104364.48	2	0.0377	3517.4
93	LPR-A-03-2-20-20	1,380	6.44	0	338238.35	2	0.0205	6649.1
94	LPR-A-03-3-20-20	2,070	12.31	0	277194.12	10	0.0685	7222.3
95	LPR-A-03-3-10-20	2,070	12.11	0.000009	278234.22	6	0.0411	7085.6
96	LPR-A-03-4A-20-20	2,760	10.53	0	225444.46	4	0.0274	7674.9
97	LPR-A-03-4A-10-20	2,760	10.23	0	230674.05	5	0.0342	7632.8
98	LPR-A-03-4B-20-20	2,760	10.03	0	205955.13	9	0.0616	7721.6
99	LPR-A-03-4B-10-20	2,760	8.32	0	210081.05	10	0.0685	7942.3
100	LPR-A-03-4C-20-20	2,760	8.55	0.000008	282045.70	12	0.0822	8982.5
101	LPR-A-03-4C-10-20	2,760	6.51	0.00001	283933.59	10	0.0685	8827.2
102	LPR-A-03-5-20-20	3,450	9.89	0	182132.80	10	0.0685	7679.1
103	LPR-A-03-5-10-20	3,450	5.52	0	184825.89	9	0.0616	7597.0
104	LPR-A-04-4-20-20	3,696	24.32	0	291013.88	11	0.0564	9560.5
105	LPR-A-04-4-10-20	3,696	12.17	0	294378.01	10	0.0513	9722.1
106	LPR-A-04-4-20-05	3,696	22.61	0.000009	291604.11	2	0.0103	8786.1
107	LPR-A-05-4-20-02	6,088	57.30	0	711398.03	1	0.0031	14103.1
108	LPR-A-05-4-20-01	6,088	60000.00	-	-	-	-	-

	Instance	NIV	Time	Gap	Objective	$\sum_{i \in V} r_i$	$\frac{1}{\lvert V \rvert} \sum_{i \in V} r_i$	Deadhead
109	LPR-B-01-2-20-100	262	0.13	0	28318.27	1	0.0357	1416.6
110	LPR-B-01-2-20-03	262	0.11	0	28527.56	0	0.0000	1259.8
111	LPR-B-02-2-20-100	498	0.66	0	94741.95	0	0.0000	3142.4
112	LPR-B-02-2-10-100	498	0.77	0	95237.12	3	0.0566	3918.1
113	LPR-B-02-2-09-100	498	0-608	0	95238.63	4	0.0755	3663.0
114	LPR-B-02-2-08-100	498	0.62	0	95325.27	3	0.0566	3466.3
115	LPR-B-02-2-06-100	498	0.61	0	95458.07	3	0.0566	3486.8
116	LPR-B-02-2-05-100	498	0.72	0	95547.84	2	0.0377	3104.2
117	LPR-B-02-2-04-100	498	0.72	0	95680.64	2	0.0377	3124.7
118	LPR-B-02-2-01-100	498	0.69	0	95908.54	2	0.0377	3045.3
119	LPR-B-02-2-00-100	498	0.59	0	96820.48	4	0.0755	3708.3
120	LPR-B-03-4-10-100	3,124	8.91	0	249796.77	7	0.0429	8086.3
121	LPR-B-03-4-20-100	3,124	7.58	0	249074.20	7	0.0429	8297.1
122	LPR-B-03-4-20-04	3,124	12.93	0	249227.11	1	0.0061	7895.4
123	LPR-B-04-4-20-100	6,045	26.83	0	380014.97	14	0.0565	10120.2
124	LPR-B-04-4-20-05	6,045	27.22	0	380633.89	3	0.0121	8432.6
125	LPR-B-04-4-20-04	6,045	27.25	0	380785.88	2	0.0081	8282.4
126	LPR-B-04-4-20-03	6,045	28.74	0	380996.61	1	0.0040	8476.3
127	LPR-B-04-4-20-02	6,045	28.58	0	380996.61	1	0.0040	8476.3
128	LPR-B-04-4-20-01	6,045	32.04	0.000009	381357.10	0	0	8497.4
129	LPR-B-05-6-20-100	11,802	125.55	0	711108.32	22	0.0549	16270.2
130	LPR-B-05-6-20-05	11,802	126.58	0.00001	711990.07	10	0.0249	15005.6
131	LPR-B-05-6-20-02	11,802	60000.00	-	-	-	-	-
132	LPR-B-05-6-20-01	11,802	60000.00	-	-	-	-	-
133	LPR-C-01-3-20-100	393	0.13	0	17325.62	5	0.17857	1967.0
134	LPR-C-01-3-20-10	393	0.12	0	17894.65	0	0	1925.1
135	LPR-C-02-3-20-100	750	0.64	0	58836.93	10	0.1887	4523.5
136	LPR-C-02-3-20-10	750	0.66	0	59267.68	1	0.0189	3838.5
137	LPR-C-03-4-20-100	3,188	8.42	0	263035.30	11	0.0675	8354.0
138	LPR-C-03-4-20-05	3,188	7.63	0.000006	263527.73	3	0.0184	7994.3
139	LPR-C-03-4-20-04	3,188	11.06	0.000003	263760.40	1	0.0061	7419.4
140	LPR-C-03-4-20-03	3,188	8.16	0	263760.40	1	0.0061	7419.4
141	LPR-C-03-4-20-02	3,188	7.72	0	263883.61	0	0	7401.4
142	LPR-C-03-4-20-01	3,188	14.57	0.000009	263883.61	0	0	7389.7

Note: "-" means no feasible solution was reported.